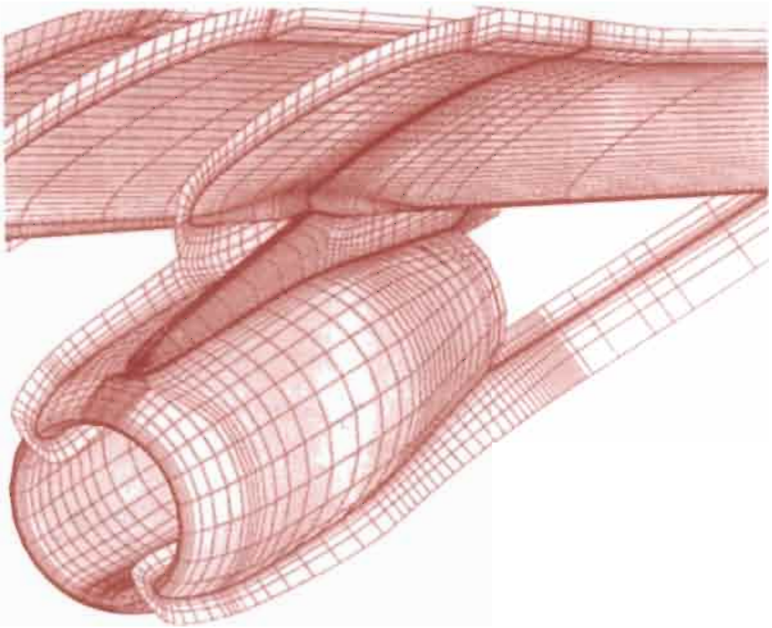
COMPUTATIONAL FLUID DYNAMICS: PRINCIPLES AND APPLICATIONS



J. Blazek

ELSEVIER

Computational Fluid Dynamics:
Principles and Applications

Elsevier Science Internet Homepage

http://www.elsevier.nl (Europe)
http://www.elsevier.com (America)
http://www.elsevier.co.jp (Asia)

Consult the Elsevier homepage for full catalogue information on all books, journals and electronic products and services.

Elsevier Titles of Related Interest

Computational Fluids and Solid Mechanics
Ed. K-J Bathe
ISBN: 008-0439446

The Mathematics of Finite Elements and Applications X
Ed. J.R. Whiteman
ISBN: 008-0435688

APCOM '99 – 4th Asia Pacific Conference on Computational Mechanics
Ed. K.H. Lee
ISBN: 008-0432093

Related Journals

Free specimen copy gladly sent on request. Elsevier Science Ltd, The Boulevard, Langford Lane, Kidlington, Oxford, OX5 1GB, UK

Advances in Engineering Software
Computer Methods in Applied Mechanics and Engineering
Computers and Fluids
Computers and Structures
Engineering Analysis with Boundary Elements
Finite Elements in Analysis and Design
International Journal of Heat and Mass Transfer
Probabilistic Engineering Mechanics

To Contact the Publisher

Elsevier Science welcomes enquiries concerning publishing proposals: books, journal special issues, conference proceedings, etc. All formats and media can be considered. Should you have a publishing proposal you wish to discuss, please contact, without obligation, the publisher responsible for Elsevier's numerical methods in engineering programme:

Dr James Milne
Publisher, Engineering and Technology
Elsevier Science Ltd
The Boulevard, Langford Lane
Kidlington, Oxford
OX5 1GB, UK

Phone: +44 1865 843891
Fax: +44 1865 843920
E.mail: j.milne@elsevier.co.uk

General enquiries, including placing orders, should be directed to Elsevier's Regional Sales Offices – please access the Elsevier homepage for full contact details (homepage details at the top of this page).

Computational Fluid Dynamics: Principles and Applications

J. Blazek

Alstom Power Ltd.,
Baden-Daettwil, Switzerland

2001
ELSEVIER

Amsterdam · London · New York · Oxford · Paris · Shannon · Tokyo

ELSEVIER SCIENCE Ltd
The Boulevard, Langford Lane
Kidlington, Oxford OX5 1GB, UK

First edition 2001

Library of Congress Cataloging in Publication Data
A catalog record from the Library of Congress has been applied for.

British Library Cataloguing in Publication Data

```
Blazek, J.
  Computational fluid dynamics : principles and applications
  1.Fluid dynamics - Computer simulation 2.Fluid dynamics -
  Mathematical models
  I.Title
  532'.05

  ISBN 0080430090
```

ISBN: 0 08 043009 0

The paper used in this publication meets the requirements of ANSI/NISO Z39.48-1992 (Permanence of Paper).
Printed in The Netherlands.

Contents

Acknowledgements xi

List of Symbols xiii

Abbreviations xix

1 Introduction 1

2 Governing Equations 5
- 2.1 The Flow and its Mathematical Description . . . 5
- 2.2 Conservation Laws . . . 8
  - 2.2.1 The Continuity Equation . . . 8
  - 2.2.2 The Momentum Equation . . . 8
  - 2.2.3 The Energy Equation . . . 10
- 2.3 Viscous Stresses . . . 13
- 2.4 Complete System of the Navier-Stokes Equations . . . 16
  - 2.4.1 Formulation for a Perfect Gas . . . 18
  - 2.4.2 Formulation for a Real Gas . . . 19
  - 2.4.3 Simplifications to the Navier-Stokes Equations . . . 22
- Bibliography . . . 26

3 Principles of Solution of the Governing Equations 29
- 3.1 Spatial Discretisation . . . 32
  - 3.1.1 Finite Difference Method . . . 36
  - 3.1.2 Finite Volume Method . . . 37
  - 3.1.3 Finite Element Method . . . 39
  - 3.1.4 Other Discretisation Methods . . . 40
  - 3.1.5 Central versus Upwind Schemes . . . 41
- 3.2 Temporal Discretisation . . . 45
  - 3.2.1 Explicit Schemes . . . 46
  - 3.2.2 Implicit Schemes . . . 49
- 3.3 Turbulence Modelling . . . 53
- 3.4 Initial and Boundary Conditions . . . 56
- Bibliography . . . 58

4 Spatial Discretisation: Structured Finite Volume Schemes 75
4.1 Geometrical Quantities of a Control Volume 79
4.1.1 Two-Dimensional Case 79
4.1.2 Three-Dimensional Case 80
4.2 General Discretisation Methodologies 83
4.2.1 Cell-Centred Scheme 83
4.2.2 Cell-Vertex Scheme: Overlapping Control Volumes 85
4.2.3 Cell-Vertex Scheme: Dual Control Volumes 88
4.2.4 Cell-Centred versus Cell-Vertex Schemes 91
4.3 Discretisation of Convective Fluxes 93
4.3.1 Central Scheme with Artificial Dissipation 95
4.3.2 Flux-Vector Splitting Schemes 98
4.3.3 Flux-Difference Splitting Schemes 105
4.3.4 Total Variation Diminishing Schemes 108
4.3.5 Limiter Functions 110
4.4 Discretisation of Viscous Fluxes 116
4.4.1 Cell-Centred Scheme 118
4.4.2 Cell-Vertex Scheme 119
Bibliography 120

5 Spatial Discretisation: Unstructured Finite Volume Schemes 129
5.1 Geometrical Quantities of a Control Volume 134
5.1.1 Two-Dimensional Case 134
5.1.2 Three-Dimensional Case 135
5.2 General Discretisation Methodologies 138
5.2.1 Cell-Centred Scheme 139
5.2.2 Median-Dual Cell-Vertex Scheme 142
5.2.3 Cell-Centred versus Median-Dual Scheme 146
5.3 Discretisation of Convective Fluxes 150
5.3.1 Central Schemes with Artificial Dissipation 150
5.3.2 Upwind Schemes 154
5.3.3 Solution Reconstruction 154
5.3.4 Evaluation of Gradients 160
5.3.5 Limiter Functions 165
5.4 Discretisation of Viscous Fluxes 169
5.4.1 Element-Based Gradients 169
5.4.2 Average of Gradients 171
Bibliography 174

6 Temporal Discretisation 181
6.1 Explicit Time-Stepping Schemes 182
6.1.1 Multistage Schemes (Runge-Kutta) 182
6.1.2 Hybrid Multistage Schemes 184
6.1.3 Treatment of the Source Term 185
6.1.4 Determination of the Maximum Time Step 186
6.2 Implicit Time-Stepping Schemes 190

6.2.1 Matrix Form of Implicit Operator 191
6.2.2 Evaluation of the Flux Jacobian 195
6.2.3 ADI Scheme 199
6.2.4 LU-SGS Scheme 202
6.2.5 Newton-Krylov Method 208
6.3 Methodologies for Unsteady Flows 212
6.3.1 Dual Time-Stepping for Explicit Multistage Schemes . . . 213
6.3.2 Dual Time-Stepping for Implicit Schemes 215
Bibliography 216

7 Turbulence Modelling **225**
7.1 Basic Equations of Turbulence 228
7.1.1 Reynolds Averaging 229
7.1.2 Favre (Mass) Averaging 230
7.1.3 Reynolds-Averaged Navier-Stokes Equations 231
7.1.4 Favre- and Reynolds-Averaged Navier-Stokes Equations . 232
7.1.5 Eddy-Viscosity Hypothesis 233
7.1.6 Non-Linear Eddy Viscosity 235
7.1.7 Reynolds-Stress Transport Equation 236
7.2 First-Order Closures 238
7.2.1 Spalart-Allmaras One-Equation Model 238
7.2.2 K-ε Two-Equation Model 241
7.2.3 SST Two-Equation Model of Menter 245
7.3 Large-Eddy Simulation 248
7.3.1 Spatial Filtering 249
7.3.2 Filtered Governing Equations 250
7.3.3 Subgrid-Scale Modelling 252
7.3.4 Wall Models 255
Bibliography 256

8 Boundary Conditions **267**
8.1 Concept of Dummy Cells 268
8.2 Solid Wall 270
8.2.1 Inviscid Flow 270
8.2.2 Viscous Flow 275
8.3 Farfield 277
8.3.1 Concept of Characteristic Variables 277
8.3.2 Modifications for Lifting Bodies 279
8.4 Inlet/Outlet Boundary 283
8.5 Symmetry Plane 285
8.6 Coordinate Cut 286
8.7 Periodic Boundaries 287
8.8 Interface Between Grid Blocks 290
8.9 Flow Gradients at Boundaries of Unstructured Grids 293
Bibliography 294

9 Acceleration Techniques **299**
9.1 Local Time-Stepping 299
9.2 Enthalpy Damping 300
9.3 Residual Smoothing 301
9.3.1 Central IRS on Structured Grids 301
9.3.2 Central IRS on Unstructured Grids 303
9.3.3 Upwind IRS on Structured Grids 303
9.4 Multigrid 305
9.4.1 Basic Multigrid Cycle 306
9.4.2 Multigrid Strategies 308
9.4.3 Implementation on Structured Grids 309
9.4.4 Implementation on Unstructured Grids 315
9.5 Preconditioning for Low Mach Numbers 320
Bibliography 324

10 Consistency, Accuracy and Stability **331**
10.1 Consistency Requirements 332
10.2 Accuracy of Discretisation 333
10.3 Von Neumann Stability Analysis 334
10.3.1 Fourier Symbol and Amplification Factor 334
10.3.2 Convection Model Equation 335
10.3.3 Convection-Diffusion Model Equation 336
10.3.4 Explicit Time-Stepping 337
10.3.5 Implicit Time-Stepping 343
10.3.6 Derivation of the CFL Condition 347
Bibliography 350

11 Principles of Grid Generation **353**
11.1 Structured Grids 356
11.1.1 C-, H-, and O-Grid Topology 357
11.1.2 Algebraic Grid Generation 359
11.1.3 Elliptic Grid Generation 363
11.1.4 Hyperbolic Grid Generation 365
11.2 Unstructured Grids 367
11.2.1 Delaunay Triangulation 368
11.2.2 Advancing-Front Method 373
11.2.3 Generation of Anisotropic Grids 374
11.2.4 Mixed-Element/Hybrid Grids 379
11.2.5 Assessment and Improvement of Grid Quality 381
Bibliography 384

12 Description of the Source Codes **393**
12.1 Programs for Stability Analysis 395
12.2 Structured 1-D Grid Generator 395
12.3 Structured 2-D Grid Generators 396
12.4 Structured to Unstructured Grid Converter 396

12.5 Quasi 1-D Euler Solver 396
12.6 Structured 2-D Euler Solver 398
12.7 Unstructured 2-D Euler Solver 400
Bibliography 400

A Appendix **401**
A.1 Governing Equations in Differential Form 401
A.2 Mathematical Character of the Governing Equations 407
A.2.1 Hyperbolic Equations 407
A.2.2 Parabolic Equations 409
A.2.3 Elliptic Equations 409
A.3 Navier-Stokes Equations in Rotating Frame of Reference 411
A.4 Navier-Stokes Equations Formulated for Moving Grids 414
A.5 Thin Shear Layer Approximation 416
A.6 Parabolised Navier-Stokes Equations 418
A.7 Convective Flux Jacobian 419
A.8 Viscous Flux Jacobian 421
A.9 Transformation from Conservative to Characteristic Variables . . 424
A.10 GMRES Algorithm 427
A.11 Tensor Notation 431
Bibliography 432

Index **435**

Acknowledgements

First of all I would like to thank my father for the initial motivation to start this project, as well as for his continuous help with the text and especially with the drawings. I thank my former colleagues from the Institute of Design Aerodynamics at the DLR in Braunschweig, Germany Norbert Kroll, Cord Rossow, Jose Longo, Rolf Radespiel and others for the opportunity to learn a lot about CFD and for the stimulating atmosphere. I also thank my colleague Andreas Haselbacher from ALSTOM Power in Daettwil, Switzerland (now at the University of Illinois at Urbana-Champaign) for reading and correcting significant parts of the manuscript, as well as for many fruitful discussions. I gratefully acknowledge the help of Olaf Brodersen from the DLR in Brauschweig and of Dimitri Mavriplis from ICASE, who provided several pictures of surface grids of transport aircraft configurations.

List of Symbols

$\bar{A}_c$	Jacobian of convective fluxes
$\bar{A}_v$	Jacobian of viscous fluxes
b	constant depth of control volume in two dimensions
c	speed of sound
c_p	specific heat coefficient at constant pressure
c_v	specific heat coefficient at constant volume
$\vec{C}$	vector of characteristic variables
C_m	molar concentration of species m $(= \rho Y_m / W_m)$
C_S	Smagorinsky constant
d	distance
$\mathbf{D}$	diagonal part of implicit operator
$\vec{D}$	artificial dissipation
D_m	effective binary diffusivity of species m
e	internal energy per unit mass
E	total energy per unit mass
f	Fourier symbol of the time-stepping operator
$\vec{f}_e$	external force vector
$\vec{F}$	flux vector
$\overline{\overline{F}}$	flux tensor
g	amplification factor
h	enthalpy
H	total (stagnation) enthalpy
$\bar{H}$	Hessian matrix (matrix of second derivatives)
I	imaginary unit $(I = \sqrt{-1})$
$\bar{I}$	identity matrix
$\overline{\overline{I}}$	unit tensor
$\hat{I}_h^{2h}$	interpolation operator

I_h^{2h}	restriction operator
I_{2h}^h	prolongation operator
$\bar{J}$	system matrix (implicit operator)
J^{-1}	inverse of determinant of coordinate transformation Jacobian
k	thermal conductivity coefficient
K	turbulent kinetic energy
K_f, K_b	forward and backward reaction rate constants
l_T	turbulent length scale
$\mathbf{L}$	strictly lower part of implicit operator
L_{ij}	components of Leonard stress tensor
M	Mach number
$\bar{M}$	mass matrix
$\vec{n}$	unit normal vector (outward pointing) of control volume face
n_x, n_y, n_z	components of the unit normal vector in $x-, y-, z-$direction
N	number of grid points, cells, or control volumes
N_A	number of adjacent control volumes
N_F	number of control volume faces
p	static pressure
$\bar{P}$	transformation matrix from conservative to primitive variables
Pr	Prandtl number
$\dot{q}_h$	heat flux due to radiation, chemical reactions, etc.
Q	source term
$\vec{r}$	position vector (Cartesian coordinates); residual (GMRES)
$\vec{r}_{ij}$	vector from point i to point j
R	specific gas constant
R_u	universal gas constant (= 8314.34 J/kg-mole K)
$\vec{R}$	residual, right-hand side
$\vec{R}^*$	smoothed residual
$\bar{\mathcal{R}}$	rotation matrix
Re	Reynolds number
$\dot{s}_m$	rate of change of species m due to chemical reactions
$\vec{S}$	face vector ($= \vec{n}\,\Delta S$)
S_{ij}	components of strain-rate tensor
S_x, S_y, S_z	Cartesian components of the face vector
dS	surface element
ΔS	length / area of a face of a control volume

t	time
t_T	turbulent time scale
Δt	time step
T	static temperature
$\bar{T}$	matrix of right eigenvectors
$\bar{T}^{-1}$	matrix of left eigenvectors
u, v, w	Cartesian velocity components
u_τ	skin friction velocity ($= \sqrt{\tau_w/\rho}$)
U	general (scalar) flow variable
$\mathbf{U}$	strictly upper part of implicit operator
$\vec{U}$	vector of general flow variables
$\vec{v}$	velocity vector with the components u, v, and w
V	contravariant velocity
V_r	contravariant velocity relative to grid motion
V_t	contravariant velocity of control volume face
$\text{curl}\,\vec{v}$	curl of $\vec{v}$ $\left(= \vec{\nabla} \times \vec{v} = \left[\frac{\partial w}{\partial y} - \frac{\partial v}{\partial z}, \frac{\partial u}{\partial z} - \frac{\partial w}{\partial x}, \frac{\partial v}{\partial x} - \frac{\partial u}{\partial y}\right]\right)$
$\text{div}\,\vec{v}$	divergence of $\vec{v}$ $\left(= \vec{\nabla} \cdot \vec{v} = \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z}\right)$
W_m	molecular weight of species m
$\vec{W}$	vector of conservative variables ($= [\rho, \rho u, \rho v, \rho w, \rho E]^T$)
$\vec{W}_p$	vector of primitive variables ($= [p, u, v, w, T]^T$)
x, y, z	Cartesian coordinate system
Δx	cell size in x-direction
y^+	non-dimensional wall coordinate ($= \rho y u_\tau / \mu_w$)
Y_m	mass fraction of species m
z	Fourier symbol of the spatial operator
α	angle of attack, inlet angle
α_m	coefficient of the Runge-Kutta scheme (in stage m)
β	parameter to control time accuracy of an implicit scheme
β_m	blending coefficient (in stage m of the Runge-Kutta scheme)
γ	ratio of specific heat coefficients at constant pressure and volume
Γ	circulation
$\bar{\Gamma}$	preconditioning matrix
δ_{ij}	Kronecker symbol
ε	rate of turbulent energy dissipation

ϵ	smoothing coefficient (implicit residual smoothing); parameter
κ	thermal diffusivity coefficient
λ	second viscosity coefficient
Λ_c	eigenvalue of convective flux Jacobian
$\bar{\Lambda}_c$	diagonal matrix of eigenvalues of convective flux Jacobian
$\hat{\Lambda}_c$	spectral radius of convective flux Jacobian
$\hat{\Lambda}_v$	spectral radius of viscous flux Jacobian
μ	dynamic viscosity coefficient
ν	kinematic viscosity coefficient ($= \mu/\rho$)
ξ, η, ζ	curvilinear coordinate system
ρ	density
σ	Courant-Friedrichs-Lewy (CFL) number
σ^*	CFL number due to residual smoothing
τ	viscous stress
τ_w	wall shear stress
$\bar{\bar{\tau}}$	viscous stress tensor (normal and shear stresses)
τ_{ij}	components of viscous stress tensor
τ_{ij}^F	components of Favre-averaged Reynolds stress tensor
τ_{ij}^R	components of Reynolds stress tensor
τ_{ij}^S	components of subgrid-scale stress tensor
τ_{ij}^{SF}	components of Favre-filtered subgrid-scale stress tensor
τ_{ij}^{SR}	components of subgrid-scale Reynolds stress tensor
ω	rate of dissipation per unit turbulent kinetic energy ($=\varepsilon/K$)
Υ	pressure sensor
Ω	control volume
Ω_{ij}	components of rotation-rate tensor
$\partial\Omega$	boundary of a control volume
Ψ	limiter function
$\vec{\nabla} U$	gradient of scalar U $\left(= \left[\frac{\partial U}{\partial x}, \frac{\partial U}{\partial y}, \frac{\partial U}{\partial z}\right]\right)$
$\nabla^2 U$	Laplace of scalar U $\left(= \frac{\partial^2 U}{\partial x^2} + \frac{\partial^2 U}{\partial y^2} + \frac{\partial^2 U}{\partial z^2}\right)$
$\lVert\vec{U}\rVert_2$	2-norm of vector $\vec{U}$ $\left(= \sqrt{\vec{U} \cdot \vec{U}}\right)$

Subscripts

C	convective part
c	related to convection
D	diffusive part
i, j, k	nodal point index
I, J, K	index of a control volume
L	laminar; left
m	index of control volume face; species
R	right
T	turbulent
v	viscous part
V	related to volume
w	wall
x, y, z	components in the x-, y-, z-direction
∞	at infinity (farfield)

Superscripts

I, J, K	direction in computational space
n	previous time level
$n+1$	new time level
T	transpose
$\tilde{}$	Favre averaged mean value; Favre-filtered value (LES)
$''$	fluctuating part of Favre decomposition; subgrid scale (LES)
$\bar{}$	Reynolds averaged mean value; filtered value (LES)
$'$	fluctuating part of Reynolds decomposition; subgrid scale (LES)

Abbreviations

AIAA	American Institute of Aeronautics and Astronautics
AGARD	Advisory Group for Aerospace Research and Development (NATO)
ARC	Aeronautical Research Council, UK
ASME	The American Society of Mechanical Engineers
CERCA	Centre de Recherche en Calcul Applique (Centre for Research on Computation and its Applications), Montreal, Canada
CERFACS	Centre Europeen de Recherche et de Formation Avancee en Calcul Scientifique (European Centre for Research and Advanced Training in Scientific Computation), France
DFVLR	(now DLR) Deutsche Forschungs- und Versuchsanstalt für Luft- und Raumfahrt (German Aerospace Research Establishment)
DLR	Deutsches Zentrum für Luft- und Raumfahrt (German Aerospace Center)
ERCOFTAC	European Research Community on Flow, Turbulence and Combustion
ESA	European Space Agency
FFA	Flygtekniska Försöksanstalten (The Aeronautical Research Institute of Sweden)
GAMM	Gesellschaft für Angewandte Mathematik und Mechanik (German Society of Applied Mathematics and Mechanics)
ICASE	Institute for Computer Applications in Science and Engineering, NASA Langley Research Center, Hampton, VA, USA
INRIA	Institut National de Recherche en Informatique et en Automatique (The French National Institute for Research in Computer Science and Control)
ISABE	International Society for Air Breathing Engines

MAE	Department of Mechanical and Aerospace Engineering, Princeton University, Princeton, USA
NACA	(now NASA) The National Advisory Committee for Aeronautics, USA
NASA	National Aeronautics and Space Administration, USA
NLR	Nationaal Lucht en Ruimtevaartlaboratorium (National Aerospace Laboratory), The Netherlands
ONERA	Office National d'Etudes et de Recherches Aerospatiales (National Institute for Aerospace Studies and Research), France
SIAM	Society of Industrial and Applied Mathematics, USA
VKI	Von Karman Institute for Fluid Dynamics, Belgium
ZAMM	Zeitschrift für angewandte Mathematik und Mechanik (Journal of Applied Mathematics and Mechanics), Germany
ZFW	Zeitschrift für Flugwissenschaften und Weltraumforschung (Journal of Aeronautics and Space Research), Germany
1D	one dimension
1-D	one-dimensional
2D	two dimensions
2-D	two-dimensional
3D	three dimensions
3-D	three-dimensional

Chapter 1

Introduction

The history of Computational Fluid Dynamics, or CFD for short, started in the early 1970's. Around that time, it became an acronym for a combination of physics, numerical mathematics, and, to some extent, computer sciences employed to simulate fluid flows. The beginning of CFD was triggered by the availability of increasingly more powerful mainframes and the advances in CFD are still tightly coupled to the evolution of computer technology. Among the first applications of the CFD methods was the simulation of transonic flows based on the solution of the non-linear potential equation. With the beginning of the 1980's, the solution of first two-dimensional (2-D) and later also three-dimensional (3-D) Euler equations became feasible. Thanks to the rapidly increasing speed of supercomputers and due to the development of a variety of numerical acceleration techniques like multigrid, it was possible to compute inviscid flows past complete aircraft configurations or inside of turbomachines. With the mid 1980's, the focus started to shift to the significantly more demanding simulation of viscous flows governed by the Navier-Stokes equations. Together with this, a variety of turbulence models evolved with different degree of numerical complexity and accuracy. The leading edge in turbulence modelling is represented by the Direct Numerical Simulation (DNS) and the Large Eddy Simulation (LES). However, both approaches are still far away from being usable in engineering applications.

With the advances of the numerical methodologies, particularly of the implicit schemes, the solution of flow problems which require real gas modelling became also feasible by the end of 1980's. Among the first large scale application, 3-D hypersonic flow past re-entry vehicles, like the European HERMES shuttle, was computed using equilibrium and later non-equilibrium chemistry models. Many research activities were and still are devoted to the numerical simulation of combustion and particularly to flame modelling. These efforts are quite important for the development of low emission gas turbines and engines. Also the modelling of steam and in particular of condensing steam became a key for the design of efficient steam turbines.

Due to the steadily increasing demands on the complexity and fidelity of

flow simulations, grid generation methods had to become more and more sophisticated. The development started first with relatively simple structured meshes constructed either by algebraic methods or by using partial differential equations. But with increasing geometrical complexity of the configurations, the grids had to be broken into a number of topologically simpler blocks (multiblock approach). The next logical step was to allow for non-matching interfaces between the grid blocks in order to relieve the constraints put on the grid generation in a single block. Finally, solution methodologies were introduced which can deal with grids overlapping each other (Chimera technique). This allowed for example to simulate the flow past the complete Space Shuttle vehicle with external tank and boosters attached. However, the generation of a structured, multiblock grid for a complicated geometry may still take weeks to accomplish. Therefore, the research also focused on the development of unstructured grid generators (and flow solvers), which promise significantly reduced setup times, with only a minor user intervention. Another very important feature of the unstructured methodology is the possibility of solution based grid adaptation. The first unstructured grids consisted exclusively of isotropic tetrahedra, which was fully sufficient for inviscid flows governed by the Euler equations. However, the solution of the Navier-Stokes equations requires for higher Reynolds numbers grids, which are highly stretched in the shear layers. Although such grids can also be constructed from tetrahedral elements, it is advisable to use prisms or hexahedra in the viscous flow regions and tetrahedra outside. This not only improves the solution accuracy, but it also saves the number of elements, faces and edges. Thus, the memory and run-time requirements of the simulation are reduced. In fact, today there is a very strong interest in unstructured, mixed-element grids and the corresponding flow solvers.

Nowadays, CFD methodologies are routinely employed in the fields of aircraft, turbomachinery, car, and ship design. Furthermore, CFD is also applied in meteorology, oceanography, astrophysics, in oil recovery, and also in architecture. Many numerical techniques developed for CFD are used in the solution of Maxwell equations as well. Hence, CFD is becoming an increasingly important design tool in engineering and also a substantial research tool in certain physical sciences. Due to the advances in numerical solution methods and computer technology, geometrically complex cases, like those which are often encountered in turbomachinery, can be treated. Also, large scale simulations of viscous flows can be accomplished within only a few hours on today's supercomputers, even for grids consisting of dozens of millions of grid cells. However, it would be completely wrong to think that CFD represents a mature technology now, like for example structural finite element methods. No, there are still many open questions like turbulence and combustion modelling, heat transfer, efficient solution techniques for viscous flows, robust but accurate discretisation methods, etc. Also the connection of CFD with other disciplines (like structural mechanics or heat conduction) requires further research. Quite new opportunities also arise in the design optimisation by using CFD.

The objective of this book is to provide university students with a solid foundation for understanding the numerical methods employed in today's CFD and

to familiarise them with modern CFD codes by hands-on experience. The book is also intended for engineers and scientists starting to work in the field of CFD or who are applying CFD codes. The mathematics used is always connected to the underlying physics to facilitate the understanding of the matter. The text can serve as a reference handbook too. Each chapter contains an extensive bibliography, which may form the basis for further studies.

CFD methods are concerned with the solution of equations of motion of the fluid as well as with the interaction of the fluid with solid bodies. The equations of motion of an inviscid fluid (Euler equations) and of viscous fluid (Navier-Stokes equations), the so-called governing equations, are formulated in Chapter 2 in integral form. Additional thermodynamic relations for a perfect gas as well as for a real gas are also discussed. Chapter 3 deals with the principles of solution of the governing equations. The most important methodologies are briefly described and the corresponding references are included. Chapter 3 can be used together with Chapter 2 to get acquainted with the fundamental principles of CFD.

A series of different schemes was developed for an efficient solution of the Euler and the Navier-Stokes equations. A unique feature of the present book is that it deals with both structured (Chapter 4) as well as unstructured finite volume schemes (Chapter 5), because of their broad application possibilities, especially for the treatment of complex flow problems routinely encountered in industrial environment. Attention is particularly devoted to the definition of various types of control volumes together with spatial discretisation methodologies for convective and viscous fluxes. The 3-D finite volume formulations of the most popular central and upwind schemes are presented in detail.

Within the framework of the finite volume schemes, it is possible either to integrate the unsteady governing equations with respect to time (referred to as time-stepping schemes) or to solve the steady-state governing equations directly. The time-stepping can be split up into two classes. One class comprises explicit time-stepping schemes (Section 6.1), and the other consists of implicit time-stepping schemes (Section 6.2). In order to provide a more complete overview, recently developed solution methods based on the Newton-iteration as well as standard techniques like Runge-Kutta schemes are discussed.

Two qualitatively different types of viscous fluid flows are encountered in general: laminar and turbulent. The solution of the Navier-Stokes equations does not raise any fundamental difficulties in the case of laminar flows. However, the simulation of turbulent flows continues to present a significant problem as before. A relatively simple way of modelling the turbulence is offered by the so-called Reynolds-averaged Navier-Stokes equations. On the other hand, Reynolds stress models or LES allow considerably more accurate predictions of turbulent flows. In Chapter 7, various well-proven and widely applied turbulence models of varying level of complexity are presented in detail.

To take into account the specific features of a particular problem, and to obtain an unique solution of the governing equations, it is necessary to specify appropriate boundary conditions. There are basically two types of boundary conditions: physical and numerical. Chapter 8 deals with both types for different

situations like solid walls, inlet, outlet and farfield. Symmetry planes, periodic and block boundaries are treated as well.

In order to shorten the time required to solve the governing equations for complex flow problems, it is quite essential to employ numerical acceleration technique. Chapter 9 deals extensively, among others, with approaches like implicit residual smoothing and multigrid. Another important technique, which is also described in Chapter 9 is preconditioning. It allows to use the same numerical scheme for flows, where the Mach number varies between nearly zero and transonic or higher values.

Each discretisation of the governing equations introduces a certain error – the discretisation error. Several consistency requirements have to be fulfilled by the discretisation scheme in order to ensure that the solution of the discretised equations closely approximates the solution of the original equations. This problem is addressed in the first two parts of Chapter 10. Before a particular numerical solution method is implemented, it is important to know, at least approximately, how the method will influence the stability and the convergence behaviour of the CFD code. It was frequently confirmed that the Von Neumann stability analysis can provide a good assessment of the properties of a numerical scheme. Therefore, in the third part of Chapter 10 it is dealt with stability analysis for various model equations.

One of the more challenging tasks in CFD is the generation of structured or unstructured body-fitted grids around complex geometries. The grid is used to discretise the governing equations in space. The accuracy of the flow solution is therefore tightly coupled to the quality of the grid. In Chapter 11, the most important methodologies for the generation of structured as well as unstructured grids are discussed.

In order to demonstrate the practical aspects of different numerical solution methodologies, various source codes are provided on the accompanying CD-ROM. Contained are the sources of quasi 1-D Euler as well as of 2-D Euler structured and unstructured flow solvers, respectively. Furthermore, source codes of 2-D structured algebraic and elliptic grid generators are included together with a convertor from structured to unstructured grids. Additionally, two programs are provided to conduct linear stability analysis of explicit and implicit time-stepping schemes. The source codes are completed by a set of worked out examples containing the grids, the input files and the results. All source codes are written in standard FORTRAN-77. Chapter 12 describes the contents of the CD-ROM and the capabilities of the particular programs.

The present book is finalised by the Appendix and the Index. The Appendix contains the governing equations presented in differential form as well as their characteristic properties. Formulations of the governing equations in rotating frame of reference and for moving grids are discussed along with some simplified forms. Furthermore, Jacobian and transformation matrices from conservative to characteristic variables are presented for two and three dimensions. The GMRES conjugate gradient method for the solution of linear equations systems is described next. The Appendix closes with the explanation of the tensor notation.

Chapter 2

Governing Equations

2.1 The Flow and its Mathematical Description

Before we turn to the derivation of the basic equations describing the behaviour of the fluid, it may be convenient to clarify what the term '*fluid dynamics*' stands for. It is, in fact, the investigation of the interactive motion of a large number of individual particles. In our case, these are molecules or atoms. That means, we suppose the density of the fluid is high enough, so that it can be approximated as a *continuum*. It implies that even an infinitesimally small (in the sense of differential calculus) element of the fluid still contains a sufficient number of particles, for which we can specify mean velocity and mean kinetic energy. In this way, we are able to define velocity, pressure, temperature, density and other important quantities at each point of the fluid.

The derivation of the principal equations of fluid dynamics is based on the fact that the dynamical behaviour of a fluid is determined by the following *conservation laws*, namely:

1. the conservation of mass,

2. the conservation of momentum, and

3. the conservation of energy.

The conservation of a certain flow quantity means that its total variation inside an arbitrary volume can be expressed as the net effect of the amount of the quantity being transported across the boundary, any internal forces and sources, and external forces acting on the volume. The amount of the quantity crossing the boundary is called *flux*. The flux can be in general decomposed into two different parts: one due to the convective transport and the other one due to the molecular motion present in the fluid at rest. This second contribution is of a diffusive nature – it is proportional to the gradient of the quantity considered and hence it will vanish for a homogeneous distribution.

The discussion of the conservation laws leads us quite naturally to the idea of dividing the flow field into a number of volumes and to concentrate on the modelling of the behaviour of the fluid in one such finite region. For this purpose, we define the so-called *finite control volume* and try to develop a mathematical description of its physical properties.

Finite control volume

Consider a general flow field as represented by streamlines in Fig. 2.1. An arbitrary finite region of the flow, bounded by the closed surface $\partial\Omega$ and fixed in space, defines the control volume Ω. We also introduce a surface element as dS and its associated, outward pointing unit normal vector as $\vec{n}$.

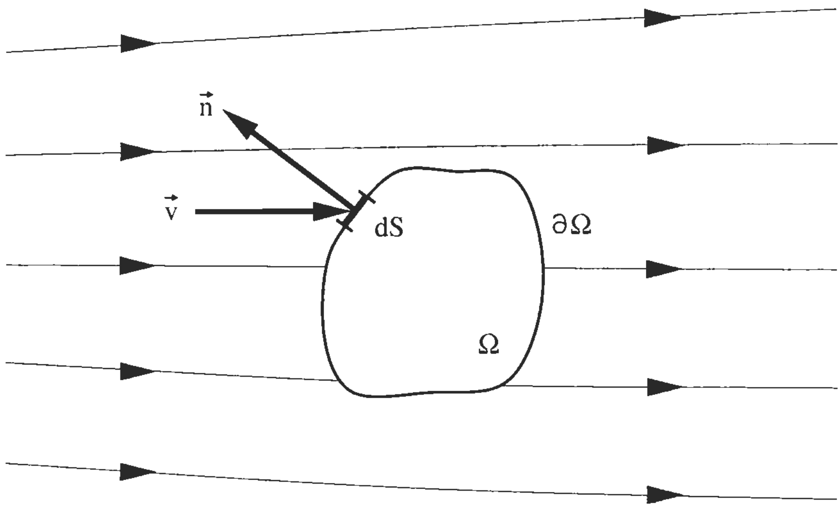


Figure 2.1: Definition of a finite control volume (fixed in space).

The conservation law applied to an exemplary scalar quantity per unit volume U now says that its variation in time within Ω, i.e.,

$$\frac{\partial}{\partial t}\int_{\Omega} U \, d\Omega$$

is equal to the sum of the contributions due to the *convective flux* - amount of the quantity U entering the control volume through the boundary with the velocity $\vec{v}$ - hence $U\vec{v}$

$$-\oint_{\partial\Omega} U(\vec{v}\cdot\vec{n})\, dS,$$

due to the *diffusive flux* – expressed by the generalised Fick's gradient law

$$\oint_{\partial\Omega} \kappa\rho\left[\nabla(U/\rho)\cdot\vec{n}\right] dS,$$

where κ is the *thermal diffusivity coefficient*, and due to the volume as well as surface sources, Q_V, $\vec{Q}_S$, i.e.,

$$\int_{\Omega} Q_V d\Omega + \oint_{\partial\Omega} (\vec{Q}_S \cdot \vec{n})\, dS\,,$$

respectively. After summing the above contributions, we obtain the following general form of the conservation law for the scalar quantity U

$$\frac{\partial}{\partial t}\int_{\Omega} U\, d\Omega + \oint_{\partial\Omega} \left[U(\vec{v}\cdot\vec{n}) - \kappa\rho\,(\nabla U^{*}\cdot\vec{n})\right] dS$$
$$= \int_{\Omega} Q_V d\Omega + \oint_{\partial\Omega} (\vec{Q}_S \cdot \vec{n})\, dS \tag{2.1}$$

where U^* denotes the quantity U per unit mass, i.e., U/ρ.

It is important to note that if the conserved quantity would be a vector instead of a scalar, the above Equation (2.1) would formally still be valid. But in difference, the convective and the diffusive flux would become tensors instead of vectors – $\overline{\overline{F}}_C$ the *convective flux tensor* and $\overline{\overline{F}}_D$ the *diffusive flux tensor*. The volume sources would be a vector $\vec{Q}_V$, and the surface sources would change into a tensor $\overline{\overline{Q}}_S$. We can therefore write the conservation law for a general vector quantity $\vec{U}$ as

$$\frac{\partial}{\partial t}\int_{\Omega} \vec{U}\, d\Omega + \oint_{\partial\Omega} \left[(\overline{\overline{F}}_C - \overline{\overline{F}}_D)\cdot\vec{n}\right] dS = \int_{\Omega} \vec{Q}_V d\Omega + \oint_{\partial\Omega} (\overline{\overline{Q}}_S \cdot \vec{n})\, dS. \tag{2.2}$$

The *integral formulation* of the conservation law, as given by the Equations (2.1) or (2.2), has two very important and desirable properties:

1. if there are no volume sources present, the variation of U depends solely on the flux across the boundary $\partial\Omega$ and **not** on any flux inside the control volume Ω;

2. this particular form remain valid in the presence of discontinuities in the flow field like shocks or contact discontinuities [1].

Because of its generality and its desirable properties, it is not surprising that the majority of CFD codes is based today on the integral form of the governing equations.

In the following section, we shall utilise the above integral form in order to derive the corresponding expressions for the three conservation laws of fluid dynamics.

2.2 Conservation Laws

2.2.1 The Continuity Equation

If we restrict our attention to single-phase fluids, the law of mass conservation expresses the fact that mass cannot be created in such a fluid system, nor can disappear from it. There is also no diffusive flux contribution to the continuity equation, since for a fluid at rest, any variation of mass would imply a displacement of fluid particles.

In order to derive the continuity equation, consider the model of a finite control volume fixed in space, as sketched in Fig. 2.1. At a point on the control surface, the flow velocity is $\vec{v}$, the unit normal vector is $\vec{n}$ and dS denotes an elemental surface area. The conserved quantity in this case is the density ρ. For the time rate of change of the total mass inside the finite volume Ω we have

$$\frac{\partial}{\partial t}\int_{\Omega} \rho \, d\Omega .$$

The mass flow of a fluid through some surface fixed in space equals to the product of (density) $\times$ (surface area) $\times$ (velocity component perpendicular to the surface). Therefore, the contribution from the convective flux across each surface element dS becomes

$$\rho\,(\vec{v}\cdot\vec{n})\,dS .$$

Since by convection $\vec{n}$ always points out of the control volume, we speak of *inflow* if the product $(\vec{v}\cdot\vec{n})$ is negative, and of *outflow* if it is positive and hence the mass flow leaves the control volume.

As stated above, there are no volume or surface sources present. Thus, by taking into account the general formulation of Eq. (2.1), we can write

$$\frac{\partial}{\partial t}\int_{\Omega} \rho \, d\Omega + \oint_{\partial\Omega} \rho\,(\vec{v}\cdot\vec{n})\,dS = 0. \tag{2.3}$$

This represents the integral form of the continuity equation – the conservation law of mass.

2.2.2 The Momentum Equation

We may start the derivation of the momentum equation by recalling the particular form of Newton's second law which states that the variation of momentum is caused by the net force acting on an mass element. For the momentum of an infinitesimally small portion of the control volume Ω (see Fig. 2.1) we have

$$\rho\vec{v}\,d\Omega .$$

The variation in time of momentum within the control volume equals

$$\frac{\partial}{\partial t}\int_{\Omega} \rho\vec{v}\,d\Omega .$$

Hence, the conserved quantity is here the product of density times the velocity, i.e.,

$$\rho\vec{v} = [\,\rho u,\ \rho v,\ \rho w\,]^T.$$

The convective flux tensor, which describes the transfer of momentum across the boundary of the control volume, consists in the Cartesian coordinate system of the following three components

$$\begin{aligned} x\text{-component}: &\quad \rho u\,\vec{v} \\ y\text{-component}: &\quad \rho v\,\vec{v} \\ z\text{-component}: &\quad \rho w\,\vec{v}. \end{aligned}$$

The contribution of the convective flux tensor to the conservation of momentum is then given by

$$-\oint_{\partial\Omega} \rho\vec{v}\,(\vec{v}\cdot\vec{n})\,dS.$$

The diffusive flux is zero, since there is no diffusion of momentum possible for a fluid at rest. So, the remaining question is now, what are the forces the fluid element is exposed to? We can identify two kinds of forces acting on the control volume:

1. *External volume* or *body* forces, which act directly on the mass of the volume. These are for example gravitational, buoyancy, Coriolis or centrifugal forces. In some cases, there can be electromagnetic forces present as well.

2. *Surface* forces, which act directly on the surface of the control volume. They result from only two sources:

   (a) the pressure distribution, imposed by the outside fluid surrounding the volume,

   (b) the shear and normal stresses, resulting from the friction between the fluid and the surface of the volume.

From the above, we can see that the body force per unit volume, denoted as $\rho\vec{f}_e$, corresponds to the volume sources in Eq. (2.1). Thus, the contribution of the body (external) force to the momentum conservation is

$$\int_\Omega \rho\vec{f}_e\,d\Omega.$$

The surface sources consist then of two parts – an isotropic pressure component and a *viscous stress* tensor $\overline{\overline{\tau}}$ (for tensors see, e.g., [2]), i.e.,

$$\overline{\overline{Q}}_S = -p\overline{\overline{I}} + \overline{\overline{\tau}} \tag{2.4}$$

with $\overline{\overline{I}}$ being the unit tensor. The effect of the surface sources on the control volume is sketched in Fig. 2.2. In Section 2.3, we shall elaborate the form of

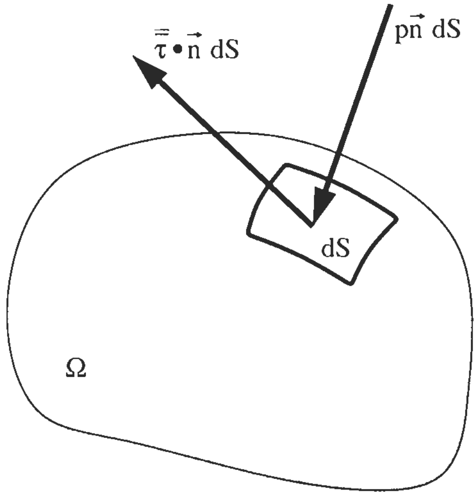


Figure 2.2: Surface forces acting on a surface element of the control volume.

the stress tensor in more detail, and in particular show how normal and shear stresses are connected to the flow velocity.

Hence, if we now sum up all the above contributions according to the general conservation law (Eq. (2.2)), we finally obtain the expression

$$\frac{\partial}{\partial t}\int_{\Omega} \rho\vec{v}\, d\Omega + \oint_{\partial\Omega} \rho\vec{v}\,(\vec{v}\cdot\vec{n})\, dS$$
$$= \int_{\Omega} \rho\vec{f}_e\, d\Omega - \oint_{\partial\Omega} p\vec{n}\, dS + \oint_{\partial\Omega} (\overline{\overline{\tau}}\cdot\vec{n})\, dS \qquad (2.5)$$

for the momentum conservation inside an arbitrary control volume Ω which is fixed in space.

2.2.3 The Energy Equation

The underlying principle that we will apply in the derivation of the energy equation, is the first law of thermodynamics. Applied to the control volume displayed in Fig. 2.1, it states that any changes in time of the total energy inside the volume are caused by the rate of work of forces acting on the volume and by the net heat flux into it. The total energy per unit mass E of a fluid is obtained by adding its internal energy per unit mass, e, to its kinetic energy per unit mass, $|\vec{v}|^2/2$. Thus, we can write for the total energy

$$E = e + \frac{|\vec{v}|^2}{2} = e + \frac{u^2+v^2+w^2}{2}\,. \qquad (2.6)$$

The conserved quantity is in this case the total energy per unit volume, i.e., ρE. Its variation in time within the volume Ω can be expressed as

$$\frac{\partial}{\partial t}\int_{\Omega} \rho E \, d\Omega .$$

Following the discussion in course of the derivation of the general conservation law (Eq. (2.1)), we can readily specify the contribution of the convective flux as

$$-\oint_{\partial\Omega} \rho E \, (\vec{v}\cdot\vec{n}) \, dS .$$

In contrast to the continuity and the momentum equation, there is now a diffusive flux. As we have already seen, it is proportional to the gradient of the conserved quantity per unit mass (Fick's law). Since the diffusive flux $\vec{F}_D$ is defined for fluid at rest, only the internal energy becomes effective and we obtain

$$\vec{F}_D = -\gamma\rho\,\kappa\,\nabla e . \tag{2.7}$$

In the above, $\gamma = c_p/c_v$ is the ratio of specific heat coefficients, and κ denotes the *thermal diffusivity coefficient* . The diffusion flux represents one part of the heat flux into the control volume, namely the diffusion of heat due to molecular thermal conduction – heat transfer due to temperature gradients. Therefore, Equation (2.7) is in general written in the form of Fourier's law of heat conduction, i.e.,

$$\vec{F}_D = -k\,\nabla T , \tag{2.8}$$

with k standing for the *thermal conductivity coefficient* and T for the absolute static temperature.

The other part of the net heat flux into the finite control volume consists of volumetric heating due to absorption or emission of radiation, or due to chemical reactions. We will denote the heat sources – the time rate of heat transfer per unit mass – as $\dot{q}_h$. Together with the rate of work done by the body forces $\vec{f}_e$, which we have introduced for the momentum equation, it completes the volume sources

$$Q_V = \rho\vec{f}_e\cdot\vec{v} + \dot{q}_h \, . \tag{2.9}$$

The last contribution to the conservation of energy, which we have yet to determine, are the surface sources Q_S. They correspond to the time rate of work done by the pressure as well as the shear and normal stresses on the fluid element (see Fig. 2.2)

$$\vec{Q}_S = -p\vec{v} + \bar{\bar{\tau}}\cdot\vec{v} . \tag{2.10}$$

Sorting now all the above contributions and terms, we obtain for the energy conservation equation the expression

$$\begin{aligned}\frac{\partial}{\partial t}\int_{\Omega} \rho E \, d\Omega + \oint_{\partial\Omega} \rho E \, (\vec{v}\cdot\vec{n}) \, dS &= \oint_{\partial\Omega} k \, (\nabla T\cdot\vec{n}) \, dS \\ &+ \int_{\Omega} (\rho\vec{f}_e\cdot\vec{v} + \dot{q}_h) d\Omega - \oint_{\partial\Omega} p \, (\vec{v}\cdot\vec{n}) \, dS + \oint_{\partial\Omega} (\bar{\bar{\tau}}\cdot\vec{v})\cdot\vec{n} \, dS \, .\end{aligned} \tag{2.11}$$

Usually, the energy equation (2.11) is written in a slightly different form. For that purpose, we will utilise the following general relation between the total enthalpy, the total energy and the pressure

$$H = h + \frac{|\vec{v}|^2}{2} = E + \frac{p}{\rho}\,. \tag{2.12}$$

When we now gather the convective ($\rho E\,\vec{v}$) and the pressure term ($p\vec{v}$) in the energy conservation law (2.11) and apply the formula (2.12), we can finally write the energy equation in the form

$$\frac{\partial}{\partial t}\int_{\Omega} \rho E\, d\Omega + \oint_{\partial\Omega} \rho H\,(\vec{v}\cdot\vec{n})\,dS = \oint_{\partial\Omega} k\,(\nabla T\cdot\vec{n})\,dS$$
$$+ \int_{\Omega} (\rho\vec{f}_e\cdot\vec{v} + \dot{q}_h)\,d\Omega + \oint_{\partial\Omega} (\overline{\overline{\tau}}\cdot\vec{v})\cdot\vec{n}\,dS\,. \tag{2.13}$$

Herewith, we have derived integral formulations of the three conservation laws: the conservation of mass (2.3), of momentum (2.5), and of energy (2.13). In the next section, we shall work out the formulation of the normal and the shear stresses in more detail.

2.3 Viscous Stresses

The viscous stresses, which originate from the friction between the fluid and the surface of an element, are described by the stress tensor $\overline{\overline{\tau}}$. In Cartesian coordinates the general form is given by

$$\overline{\overline{\tau}} = \begin{bmatrix} \tau_{xx} & \tau_{xy} & \tau_{xz} \\ \tau_{yx} & \tau_{yy} & \tau_{yz} \\ \tau_{zx} & \tau_{zy} & \tau_{zz} \end{bmatrix} \tag{2.14}$$

The notation τ_{ij} means by convention that the particular stress component affects a plane perpendicular to the i-axis, in the direction of the j-axis. The components τ_{xx}, τ_{yy}, and τ_{zz} represent the normal stresses, the other components of $\overline{\overline{\tau}}$ stand for the shear stresses, respectively. Figure 2.3 shows the stresses for a quadrilateral fluid element. One can notice that the normal stresses (Fig. 2.3a) try to displace the faces of the element in three mutually perpendicular directions, whereas the shear stresses (Fig. 2.3b) try to shear the element.

You may now ask, how the viscous stresses are evaluated. First of all, they depend on the dynamical properties of the medium. For fluids like air or water, Isaac Newton stated that the shear stress is proportional to the velocity gradient. Therefore, medium of such a type is designated as *Newtonian fluid*. On the other hand, fluids like for example melted plastic or blood behave in a different manner – they are non-Newtonian fluids. But, for the vast majority of practical problems, where the fluid can be assumed to be Newtonian, the components of the viscous stress tensor are defined by the relations [3], [4]

$$\begin{aligned}
\tau_{xx} &= \lambda\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z}\right) + 2\mu\frac{\partial u}{\partial x} \\
\tau_{yy} &= \lambda\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z}\right) + 2\mu\frac{\partial v}{\partial y} \\
\tau_{zz} &= \lambda\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z}\right) + 2\mu\frac{\partial w}{\partial z} \\
\tau_{xy} &= \tau_{yx} = \mu\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right) \\
\tau_{xz} &= \tau_{zx} = \mu\left(\frac{\partial u}{\partial z} + \frac{\partial w}{\partial x}\right) \\
\tau_{yz} &= \tau_{zy} = \mu\left(\frac{\partial v}{\partial z} + \frac{\partial w}{\partial y}\right)
\end{aligned} \tag{2.15}$$

in which λ represents the *second viscosity* coefficient, and μ denotes the *dynamic viscosity* coefficient. For convenience, we can also define the so-called *kinematic*

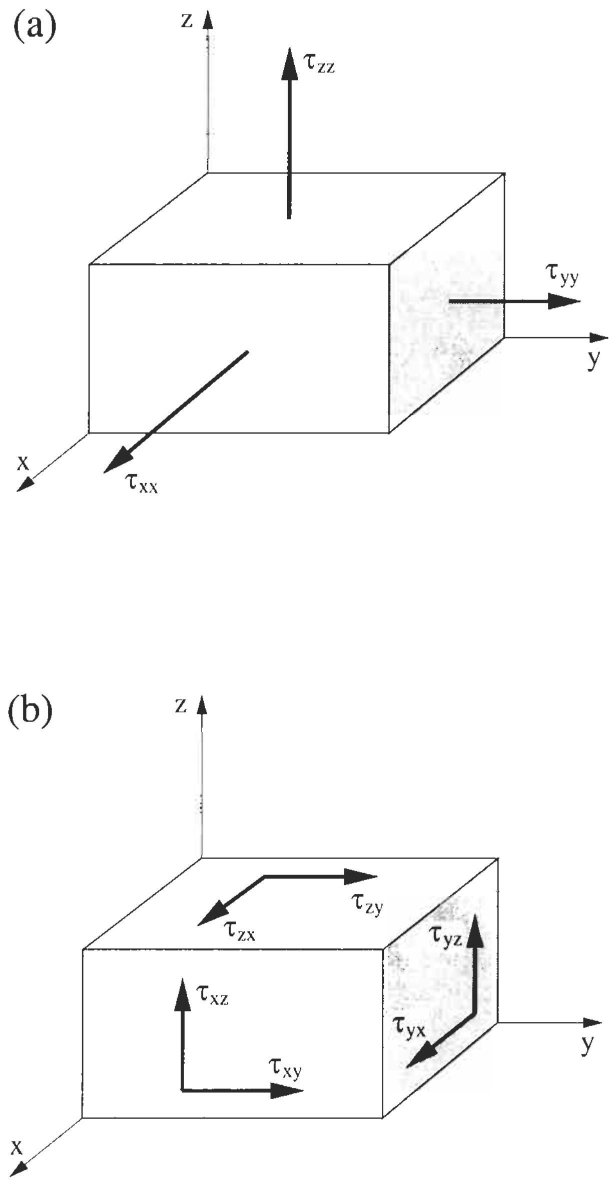


Figure 2.3: Normal (a) and shear (b) stresses acting on a fluid element.

viscosity coefficient, which is given by the formula

$$\nu = \mu/\rho \,. \tag{2.16}$$

The expressions (Eq. (2.15)) were derived by the Englishman George Stokes in the middle of the 19th century. The terms $\mu(\partial u/\partial x)$, etc. in the normal stresses (Eq. (2.15)) represent the rate of *linear* dilatation – a change in shape. On the other hand, the term $(\lambda \operatorname{div} \vec{v})$ in Eq. (2.15) represents *volumetric* dilatation – rate of change in volume, which is in essence a change in density.

In order to close the expressions for the normal stresses, Stokes introduced the hypothesis [5] that

$$\lambda + \frac{2}{3}\mu = 0 \,. \tag{2.17}$$

The above relation (2.17) is termed the ***bulk viscosity***. Bulk viscosity represents that property, which is responsible for energy dissipation in a fluid of uniform temperature during a change in volume at finite rate.

With the exception of extremely high temperatures or pressures, there is so far no experimental evidence that Stokes's hypothesis (Eq. (2.17)) does not hold (see discussion in Ref. [6]), and it is therefore used in general to eliminate λ from Eq. (2.15). Hence, we obtain for the normal viscous stresses

$$\begin{aligned}
\tau_{xx} &= 2\mu\left(\frac{\partial u}{\partial x} - \frac{1}{3}\operatorname{div}\vec{v}\right) \\
\tau_{yy} &= 2\mu\left(\frac{\partial v}{\partial y} - \frac{1}{3}\operatorname{div}\vec{v}\right) \\
\tau_{zz} &= 2\mu\left(\frac{\partial w}{\partial z} - \frac{1}{3}\operatorname{div}\vec{v}\right) .
\end{aligned} \tag{2.18}$$

It should be noted that the expressions for the normal stresses in Eq. (2.18) simplify for an incompressible fluid (constant density) because of $\operatorname{div}\vec{v} = 0$ (continuity equation).

What remains to be determined are the viscosity coefficient μ and the thermal conductivity coefficient k as functions of the state of the fluid. This can be done within the framework of continuum mechanics only on the basis of empirical assumptions. We shall return to this problem in the next section.

2.4 Complete System of the Navier-Stokes Equations

In the previous sections, we have separately derived the conservation laws of mass, momentum and energy. Now, we can collect them into one system of equations in order to obtain a better overview of the various terms involved. For this purpose, we go back to the general conservation law for a vector quantity, which is expressed by Equation (2.2). For reasons to be explained later, we will introduce two flux vectors, namely $\vec{F}_c$ and $\vec{F}_v$. The first one, $\vec{F}_c$, is related to the convective transport of quantities in the fluid. It is usually termed *vector of convective fluxes*, although for the momentum and the energy equation it also includes the pressure terms $p\vec{n}$ (Eq. (2.5)) and $p(\vec{v}\cdot\vec{n})$ (Eq. (2.11)), respectively. But, do not be confused by this. The second flux vector – *vector of viscous fluxes* $\vec{F}_v$, contains the viscous stresses as well as the heat diffusion. Additionally, let us define a source term $\vec{Q}$, which comprises all volume sources due to body forces and volumetric heating. With all this in mind and conducting the scalar product with the unit normal vector $\vec{n}$, we can cast Eq. (2.2) together with Equations (2.3), (2.5) and (2.13) into

$$\frac{\partial}{\partial t}\int_\Omega \vec{W}\,d\Omega + \oint_{\partial\Omega}(\vec{F}_c - \vec{F}_v)\,dS = \int_\Omega \vec{Q}\,d\Omega\,. \tag{2.19}$$

The vector of the so-called *conservative variables* $\vec{W}$ consists in three dimensions of the following five components

$$\vec{W} = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ \rho E \end{bmatrix}. \tag{2.20}$$

For the vector of convective fluxes we obtain

$$\vec{F}_c = \begin{bmatrix} \rho V \\ \rho uV + n_x p \\ \rho vV + n_y p \\ \rho wV + n_z p \\ \rho HV \end{bmatrix} \tag{2.21}$$

with the *contravariant velocity* V – the velocity normal to the surface element dS – being defined as the scalar product of the velocity vector and the unit normal vector, i.e.,

$$V \equiv \vec{v}\cdot\vec{n} = n_x u + n_y v + n_z w\,. \tag{2.22}$$

The total enthalpy H is given by the formula (2.12). For the vector of viscous fluxes we have with Eq. (2.14)

$$\vec{F}_v = \begin{bmatrix} 0 \\ n_x\tau_{xx} + n_y\tau_{xy} + n_z\tau_{xz} \\ n_x\tau_{yx} + n_y\tau_{yy} + n_z\tau_{yz} \\ n_x\tau_{zx} + n_y\tau_{zy} + n_z\tau_{zz} \\ n_x\Theta_x + n_y\Theta_y + n_z\Theta_z \end{bmatrix}, \tag{2.23}$$

where

$$\begin{aligned} \Theta_x &= u\tau_{xx} + v\tau_{xy} + w\tau_{xz} + k\frac{\partial T}{\partial x} \\ \Theta_y &= u\tau_{yx} + v\tau_{yy} + w\tau_{yz} + k\frac{\partial T}{\partial y} \\ \Theta_z &= u\tau_{zx} + v\tau_{zy} + w\tau_{zz} + k\frac{\partial T}{\partial z} \end{aligned} \tag{2.24}$$

are the terms describing the work of viscous stresses and the heat conduction in the fluid. Finally, the source term reads

$$\vec{Q} = \begin{bmatrix} 0 \\ \rho f_{e,x} \\ \rho f_{e,y} \\ \rho f_{e,z} \\ \rho\vec{f}_e \cdot \vec{v} + \dot{q}_h \end{bmatrix}. \tag{2.25}$$

In the case of a Newtonian fluid, i.e., if the relations Eq. (2.15) for the viscous stresses are valid, the above system of equations (Eqs. (2.19)-(2.25)) is called the *Navier-Stokes equations.* They describe the exchange (flux) of mass, momentum and energy through the boundary $\partial\Omega$ of a control volume Ω, which is fixed in space (see Fig. 2.1). We have derived the Navier-Stokes equations in integral formulation, in accordance with the conservation laws. Applying Gauss's theorem, Equation (2.19) can be re-written in differential form [7]. Since the differential form is often found in literature, it is for completeness included in the Appendix (A.1).

In some instances, for example in turbomachinery applications or geophysics, the control volume is rotating (usually steadily) about some axis. In such a case, the Navier-Stokes equations are transformed into a *rotating frame of reference.* As a consequence, the source term $\vec{Q}$ has to be extended by the effects due to the Coriolis and the centrifugal force [8]. The resulting form of the Navier-Stokes equations may be found in the Appendix (A.3). In other cases, the control volume can be subject to translation or deformation. This happens, for instance, when fluid-structure interaction is investigated. Then the Navier-Stokes equations have to be extended by a term, which describes the relative motion of the surface element dS with respect to the fixed coordinate system [9]. Additionally, the so-called *Geometric Conservation Law* (GCL) has to be fulfilled [10]-[12]. In the Appendix (A.4) we show the appropriate formulation.

The Navier-Stokes equations represent in three dimensions a system of five equations for the five conservative variables ρ, ρu, ρv, ρw, and ρE. But they contain seven unknown flow field variables, namely: ρ, u, v, w, E, p, and T. Therefore, we have to supply two additional equations, which have to be thermodynamic relations between the state variables, like for example the pressure as a function of density and temperature, and the internal energy or the enthalpy as a function of pressure and temperature. Beyond this, we have to provide the viscosity coefficient μ and the thermal conductivity coefficient k as a function of the state of the fluid, in order to close the entire system of equations. Clearly, the relationships depend on the kind of fluid being considered. In the following, we shall therefore show methods of closing the equations for two commonly encountered situations.

2.4.1 Formulation for a Perfect Gas

In pure aerodynamics, it is generally reasonable to assume that the working fluid behaves like a calorically *perfect gas*, for which the equation of state takes the form [13], [14]

$$p = \rho R T \tag{2.26}$$

where R denotes the specific gas constant. The enthalpy results from

$$h = c_p T \,. \tag{2.27}$$

It is convenient to express the pressure in terms of the conservative variables. For that purpose, we have to combine Equation (2.12), relating the total enthalpy to the total energy, together with the equation of state (2.26). Substituting expression (2.27) for the enthalpy and using the definitions

$$R = c_p - c_v \,, \quad \gamma = \frac{c_p}{c_v} \,, \tag{2.28}$$

we finally obtain for the pressure

$$p = (\gamma - 1)\,\rho \left[E - \frac{u^2 + v^2 + w^2}{2} \right] . \tag{2.29}$$

The temperature is then calculated with the help of the relationship Eq. (2.26). The coefficient of the dynamic viscosity μ is, for a perfect gas, strongly dependent on temperature but only weakly dependent on pressure. Use is frequently made of the Sutherland formula. The result for air in SI units is, for example,

$$\mu = \frac{1.45\, T^{3/2}}{T + 110} \cdot 10^{-6} \tag{2.30}$$

where the temperature T is in degree Kelvin (K). Thus, at $T = 288\,\mathrm{K}$ one obtains $\mu = 1.78 \cdot 10^{-5}\,\mathrm{kg/ms}$. The temperature dependence of the thermal

conductivity coefficient k resembles that of μ in the case of gases. By contrast, k is virtually constant in the case of liquids. For this reason, the relationship

$$k = c_p \frac{\mu}{Pr}, \tag{2.31}$$

is generally used for air. In addition, it is commonly assumed that the Prandtl number Pr is constant in the entire flow field. For air, the Prandtl number takes the value $Pr = 0.72$.

2.4.2 Formulation for a Real Gas

The matter becomes more complicated when one has to deal with a *real gas*. The reason is that now we have to model a thermodynamic process and chemical reactions in addition to the fluid dynamics. Examples for a real gas flow are the simulation of combustion, the hypersonic flow past a re-entry vehicle, or the flow in a steam turbine. In principle, two different methods can be pursued to solve the problem. The first methodology is applicable in cases, where the gas is in chemical and in thermodynamical equilibrium. This implies that there is unique equation of state. Then, the governing equations (2.19) remain unchanged. Only the values of pressure, temperature, viscosity, etc. are interpolated from lookup tables using curve fits [15], [16], [17]. But in practice, the gas is more often in chemical and/or thermodynamical non-equilibrium and has to be treated correspondingly.

Let us for illustration consider a gas mixture consisting of N different species. For a finite Damköhler number, defined as the ratio of flow-residence time to chemical-reaction time, we have to include finite-rate chemistry into our model. It has to describe the generation/destruction of species due to chemical reactions. In what follows, we will furthermore assume that the temporal and the spatial scales of fluid dynamics and chemical reactions are much larger compared to those of thermodynamics. Thus, we suppose the gas is thermodynamically in equilibrium but chemically in non-equilibrium. In order to simulate the behaviour of such a gas mixture, the Navier-Stokes equations have to be augmented by (N-1) additional transport equations for the N species [18]-[23]. Hence, we obtain formally the same system like Eq. (2.19), but now with the vectors of the conservative variables $\vec{W}$, the flux vectors $\vec{F}_c$ and $\vec{F}_v$, as well as with the source term $\vec{Q}$ extended by (N-1) species equations. Recalling the expressions (2.20) to (2.25), the vector of the conservative variables reads now

$$\vec{W} = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ \rho E \\ \rho Y_1 \\ \vdots \\ \rho Y_{N-1} \end{bmatrix}. \tag{2.32}$$

The convective and the viscous flux vectors transform into

$$\vec{F}_c = \begin{bmatrix} \rho V \\ \rho uV + n_x p \\ \rho vV + n_y p \\ \rho wV + n_z p \\ \rho HV \\ \rho Y_1 V \\ \vdots \\ \rho Y_{N-1} V \end{bmatrix}, \quad \vec{F}_v = \begin{bmatrix} 0 \\ n_x\tau_{xx} + n_y\tau_{xy} + n_z\tau_{xz} \\ n_x\tau_{yx} + n_y\tau_{yy} + n_z\tau_{yz} \\ n_x\tau_{zx} + n_y\tau_{zy} + n_z\tau_{zz} \\ n_x\Theta_x + n_y\Theta_y + n_z\Theta_z \\ n_x\Phi_{x,1} + n_y\Phi_{y,1} + n_z\Phi_{z,1} \\ \vdots \\ n_x\Phi_{x,N-1} + n_y\Phi_{y,N-1} + n_z\Phi_{z,N-1} \end{bmatrix}, \tag{2.33}$$

where

$$\begin{aligned} \Theta_x &= u\tau_{xx} + v\tau_{xy} + w\tau_{xz} + k\frac{\partial T}{\partial x} + \rho\sum_{m=1}^{N} h_m D_m \frac{\partial Y_m}{\partial x} \\ \Theta_y &= u\tau_{yx} + v\tau_{yy} + w\tau_{yz} + k\frac{\partial T}{\partial y} + \rho\sum_{m=1}^{N} h_m D_m \frac{\partial Y_m}{\partial y} \\ \Theta_z &= u\tau_{zx} + v\tau_{zy} + w\tau_{zz} + k\frac{\partial T}{\partial z} + \rho\sum_{m=1}^{N} h_m D_m \frac{\partial Y_m}{\partial z} \\ \Phi_{x,m} &= \rho D_m \frac{\partial Y_m}{\partial x} \\ \Phi_{y,m} &= \rho D_m \frac{\partial Y_m}{\partial y} \\ \Phi_{z,m} &= \rho D_m \frac{\partial Y_m}{\partial z} \,. \end{aligned} \tag{2.34}$$

Finally, the source term becomes now

$$\vec{Q} = \begin{bmatrix} 0 \\ \rho f_{e,x} \\ \rho f_{e,y} \\ \rho f_{e,z} \\ \rho \vec{f}_e \cdot \vec{v} + \dot{q}_h \\ \dot{s}_1 \\ \vdots \\ \dot{s}_{N-1} \end{bmatrix}. \tag{2.35}$$

In the above expressions (Eqs. (2.32)-(2.35)), Y_m denotes the mass fraction, h_m the enthalpy, and D_m the effective binary diffusivity of species m, respectively. Furthermore, $\dot{s}_m$ is the rate of change of species m due to chemical reactions. Note that the total density ρ of the mixture is equal to the sum of the densities of the species ρY_m. Therefore, since the total density is regarded as an independent quantity, there are only (N-1) independent densities ρY_m left. The remaining

mass fraction Y_N is obtained from

$$Y_N = 1 - \sum_{m=1}^{N-1} Y_m \,. \tag{2.36}$$

In order to find an expression for the pressure p, we first assume that the individual species behave like ideal gases, i.e.,

$$p_m = \rho Y_m \frac{R_u}{W_m} T \,, \tag{2.37}$$

with R_u denoting the universal gas constant and W_m being the molecular weight, respectively. Together with Dalton's law,

$$p = \sum_{m=1}^{N} p_m \,, \tag{2.38}$$

we can write

$$p = \rho R_u T \sum_{m=1}^{N} \frac{Y_m}{W_m} \,. \tag{2.39}$$

It is important to notice that because the gas is in thermodynamical equilibrium, all species possess the same temperature T. The temperature has to be calculated iteratively from the expression [21], [24]

$$e = \sum_{m=1}^{N} \left[Y_m \left(h_{f,m}^0 + \int_{T_{ref}}^{T} c_{p,m} \, dT \right) \right] - \frac{p}{\rho} \,. \tag{2.40}$$

The internal energy of the gas mixture e is obtained from Eq. (2.6). The quantities $h_{f,m}^0$, $c_{p,m}$, and T_{ref} denote the heat of formation, the specific heat at constant pressure, and the reference temperature, respectively, of the m-th species. Values of the above quantities as well as of the thermal conductivity k and of the dynamic viscosity μ of the species are determined from curve fits [19], [21], [23].

The last part, which remains to be modelled, is the chemical source term $\dot{s}_m$ in Eq. (2.35). The rate equations for a set of N_R elementary reactions involving N species can be written in the general form

$$\sum_{m=1}^{N} \nu_{lm}' C_m \underset{K_{bl}}{\overset{K_{fl}}{\rightleftharpoons}} \sum_{m=1}^{N} \nu_{lm}'' C_m \qquad \text{for} \quad l = 1, 2, \cdots, N_R \,. \tag{2.41}$$

In the above Eq. (2.41), ν_{lm}' and ν_{lm}'' are the stoichiometric coefficients for species m in the l-th forward and backward reaction, respectively. Furthermore, C_m stands for the molar concentration of species m ($C_m = \rho Y_m / W_m$), and finally K_{fl} and K_{bl}, respectively, denote the forward and the backward reaction rate

constants for the l-th reaction step. They are given by the empirical Arrhenius formulae

$$K_f = A_f\, T^{B_f} \exp(-E_f\,/R_u T)$$
$$K_b = A_b\, T^{B_b} \exp(-E_b\,/R_u T)\,, \tag{2.42}$$

where A_f and A_b are the Arrhenius coefficients, E_f and E_b represent the activation energies, and B_f as well as B_b are constants, respectively. The rate of change of molar concentration of species m by the l-th reaction is given by

$$\dot{C}_{lm} = (\nu''_{lm} - \nu'_{lm}) \left(K_{fl} \prod_{n=1}^{N} C_n^{\nu'_{ln}} - K_{bl} \prod_{n=1}^{N} C_n^{\nu''_{ln}} \right) . \tag{2.43}$$

Hence, together with Eq. (2.43) we can calculate the total rate of change of species m from

$$\dot{s}_m = W_m \sum_{l=1}^{N_R} \dot{C}_{lm} \,. \tag{2.44}$$

More details can be found in the references cited above. A detailed overview of the equations governing a chemically reacting flow, together with the Jacobian matrices of the fluxes and their eigenvalues, can also be found in [24].

Another practical example of real gas is the simulation of steam or, which is more demanding, of wet steam in turbomachinery applications [25]-[32]. In the later case, where the steam is mixed with water droplets, so that we speak of *multiphase flow*, it is either possible to solve an additional set of transport equations, or to trace the water droplets along a number of streamlines. These simulations have very important applications in the design of modern steam turbine cascades. The analysis of flow past turbine blades can for instance help to understand the occurrence of supercritical shocks by condensation and of flow instabilities, responsible for an additional dynamic load on the bladings resulting in loss of efficiency.

2.4.3 Simplifications to the Navier-Stokes Equations

Thin Shear Layer Approximation

When calculating flows around bodies for high Reynolds numbers (i.e., thin boundary layer with respect to a characteristic dimension), the Navier-Stokes equations (2.19) can be simplified. One necessary condition is that there is no large area of separated boundary layer. It can then be anticipated that only the gradients of the flow quantities in the normal direction to the surface of the body (η-direction in Fig. 2.4) contribute to the viscous stresses [33], [34]. On the other hand, the gradients in the other coordinate directions (ξ in Fig. 2.4)

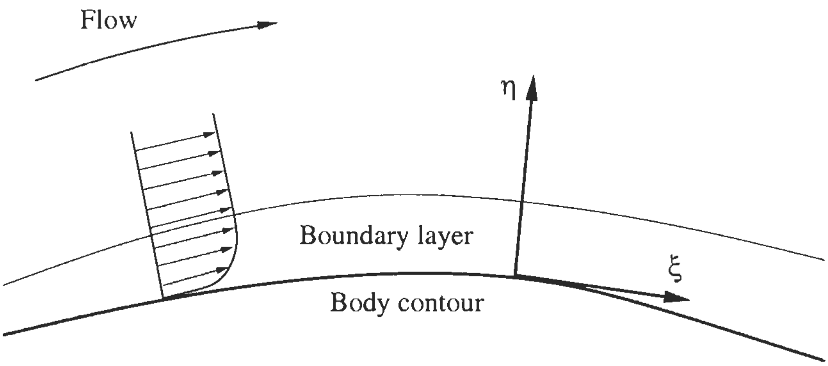


Figure 2.4: Representation of a thin boundary layer.

are neglected in the evaluation of the shear stress tensor (Eqs. (2.14, 2.15)). We speak here of the so-called *Thin Shear Layer* (TSL) approximation of the Navier-Stokes equations. The motivation for the TSL modification is that the numerical evaluation of the viscous terms becomes computationally less expensive, but, within the assumptions, the solution remains sufficiently accurate. The TSL approximation can also be justified from a practical side. In the case of high Reynolds number flows, the grid has to be very fine in the wall normal direction in order to resolve the boundary layer properly. Because of the limited computer memory and speed, much coarser grid has to be generated in the other directions. This in turn results in significantly lower numerical accuracy of the gradient evaluation compared to the normal direction. The TSL equations are for completeness presented in the Appendix (A.5). Due to the fact that secondary flow (e.g., like in a blade row) cannot be resolved appropriately, the TSL simplification is usually applied only in external aerodynamics.

Parabolised Navier-Stokes Equations

In cases, where the following three conditions are fulfilled:

- the flow is steady (i.e. $\partial \vec{W}/\partial t$=0),
- the fluid moves predominantly in one main direction (e.g., there must be no boundary layer separation),
- the cross-flow components are negligible,

the governing equations (2.19)) can be simplified to a form called the *Parabolised Navier-Stokes* (PNS) equations [8], [35]-[37]. The above conditions allow us to set the derivatives of u, v, and w with respect to the streamwise direction to

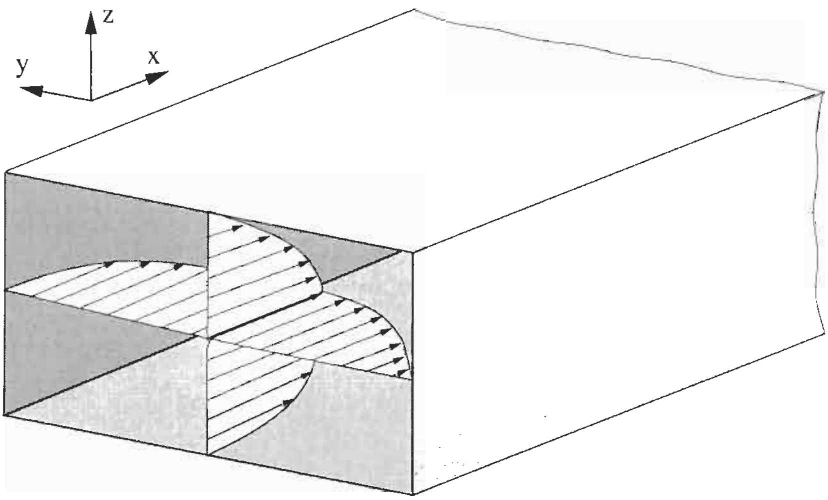


Figure 2.5: Internal flow in a duct – parabolised Navier-Stokes equations.

zero in the viscous stress terms (Eq. (2.15)). Furthermore, the components of the viscous stress tensor $\overline{\overline{\tau}}$, of the work performed by it $(\overline{\overline{\tau}} \cdot \vec{v})$, and of the heat conduction $k\nabla T$ in the streamwise direction are dropped from the viscous flux vector in Eq. (2.23). The continuity equation, as well as the convective fluxes (Eq. (2.21)) remain unchanged. For details, the reader is referred to the Appendix (A.6). Considering the situation sketched in Fig. 2.5, where the main flow direction coincides with the x coordinate, it can be shown that the PNS approximation leads to a mixed set of parabolic / elliptic equations. Namely, the momentum equation in the flow direction becomes parabolic together with the energy equation, and hence they can be solved by marching in the x-direction. The momentum equations in the y- and in the z-direction are elliptic and they have to be solved iteratively in each x-plane. Thus, the main benefit of the PNS approach is in the largely reduced complexity of the flow solution – from a complete 3-D field to a sequence of 2-D problems. A typical application of the parabolised Navier-Stokes equations is the calculation of internal flows in ducts and in pipes and also the simulation of steady supersonic flows using the space-marching method [38]-[41].

Euler Equations

As we have seen, the Navier-Stokes equations describe the behaviour of a viscous fluid. In many instances, it is a valid approximation to neglect the viscous effects completely, like for example for high Reynolds-number flows, where a boundary layer is very thin compared to the dimensions of the body. In such case, we can simply omit the vector of viscous fluxes, $\vec{F}_v$, from the Equations

(2.19). Those will then reduce to

$$\frac{\partial}{\partial t}\int_{\Omega} \vec{W}\, d\Omega + \oint_{\partial\Omega} \vec{F}_c\, dS = \int_{\Omega} \vec{Q}\, d\Omega\,. \tag{2.45}$$

The remaining terms are given by the same relations (2.20)-(2.22) and Eq. (2.25) as before. This simplified form of the governing equations is called the *Euler equations*. They describe the pure convection of flow quantities in an inviscid fluid. If the Euler equations are formulated in conservative way (like above), they allow for accurate representation of such important phenomena like shocks, expansion waves and vortices over delta wings (with sharp leading edges). Furthermore, the Euler equations served in the past – and still do – as the basis for the development of discretisation methods and boundary conditions.

However, it should be noted that today, due to the computational power of even workstations and due to the increased demands on the quality of the simulations, the Euler equations are increasingly less employed for flow computations.

Bibliography

[1] Lax, P.D.: *Weak Solutions of Nonlinear Hyperbolic Equations and their Numerical Computation.* Comm. Pure and Applied Mathematics, 7 (1954), pp. 159-193.

[2] Aris, R.: *Vectors, Tensors and the Basic Equations of Fluid Mechanics.* Dover Publ. Inc., New York, 1989.

[3] Schlichting, H.: *Boundary Layer Theory.* 7th edition, McGraw-Hill, New York, 1979.

[4] White, F.M.: *Viscous Fluid Flow.* McGraw-Hill, New York, 1991.

[5] Stokes, G.G.: *On the Theories of Internal Friction of Fluids in Motion.* Trans. Cambridge Phil. Soc., 8 (1845), pp. 287-305.

[6] Gad-el-Hak, M.: *Questions in Fluid Mechanics: Stokes' Hypothesis for a Newtonian, Isotropic Fluid.* J. of Fluids Engineering, 117 (1995), pp. 3-5.

[7] Vinokur, M.: *Conservation Equations of Gas Dynamics in Curvilinear Coordinate Systems.* J. Computational Physics, 14 (1974), pp. 105-125.

[8] Hirsch, C.: *Numerical Computation of Internal and External Flows.* Vols. 1 and 2, John Wiley and Sons, 1988.

[9] Pulliam, T.H.; Steger, J.L.: *Recent Improvements in Efficiency, Accuracy, and Convergence for Implicit Approximate Factorization Algorithms.* AIAA Paper 85-0360, 1985.

[10] Thomas, P.D.; Lombard, C.K.: *Geometric Conservation Law and Its Application to Flow Computations on Moving Grids.* AIAA Journal, 17 (1979), pp. 1030-1037.

[11] Lesoinne, M.; Farhat, C.: *Geometric Conservation Laws for Flow Problems with Moving Boundaries and Deformable Meshes and their Impact on Aeroelastic Computations.* AIAA Paper 95-1709, 1995; also in Comp. Meth. Appl. Mech. Eng., 134 (1996), pp. 71-90.

[12] Guillard, H.; Farhat, C.: *On the Significance of the GCL for Flow Computations on Moving Meshes.* AIAA Paper 99-0793, 1999.

[13] Zierep, J.: *Vorlesungen über theoretische Gasdynamik (Lectures on Theoretical Gas Dynamics).* G. Braun Verlag, Karlsruhe, 1963.

[14] Liepmann, HG.W.; Roshko, A.: *Elements of Gas Dynamics.* John Wiley & Sons, New York, 1957.

[15] Srinivasan S.; Weilmuenster, K.J: *Simplified Curve Fits for the Thermodynamic Properties of Equilibrium Air.* NASA RP-1181, 1987.

[16] Mundt, Ch.; Keraus, R.; Fischer, J.: *New, Accurate, Vectorized Approximations of State Surfaces for the Thermodynamic and Transport Properties of Equilibrium Air.* ZFW, 15 (1991), Springer Verlag, pp. 179-184.

[17] Schmatz, M.A.: *Hypersonic Three-Dimensional Navier-Stokes Calculations for Equilibrium Gas.* AIAA Paper 89-2183, 1989.

[18] Bussing, T.R.A; Murman, E.M.: *Finite-Volume Method for the Calculation of Compressible Chemically Reacting Flows.* AIAA Journal, 26 (1988), pp. 1070-1078.

[19] Molvik, G.A.; Merkle, C.L.: *A Set of Strongly Coupled, Upwind Algorithms for Computing Flows in Chemical Non-equilibrium.* AIAA Paper 89-0199, 1989.

[20] Slomski, J.F.; Anderson, J.D.; Gorski, J.J.: *Effectiveness of Multigrid in Accelerating Convergence of Multidimensional Flows in Chemical Nonequilibrium.* AIAA Paper 90-1575, 1990.

[21] Shuen, J.S.; Liou, M.S.; van Leer, B.: *Inviscid Flux-Splitting Algorithms for Real Gases with Non-equilibrium Chemistry.* J. Computational Physics 90 (1990), pp. 371-395.

[22] Li, C.P.: *Computational Aspects of Chemically Reacting Flows.* AIAA Paper 91-1574, 1991.

[23] Shuen, J.-S.; Chen, K.-H.; Choi, Y.: *A Coupled Implicit Method for Chemical Non-equilibrium Flows at All Speeds.* J. Computational Physics, 106 (1993), pp. 306-318.

[24] Yu, S.-T.; Chang, S.-C.; Jorgenson, P.C.E.; Park, S.-J.; Lai, M.-C.: *Basic Equations of Chemically Reactive Flow for Computational Fluid Dynamics.* AIAA Paper 98-1051, 1998.

[25] Bakhtar, F.; Tochai, M.T.M.: *An Investigation of Two-Dimensional Flows of Nucleating and Wet Stream by the Time-Marching Method.* Int. J. Heat and Fluid Flow 2 (1980), pp. 5-18.

[26] Young, J.B.; Snoeck, J.: *Aerothermodynamics of Low Pressure Steam Turbines and Condensers.* Eds. Moore, M.J.; Sieverding, C.H.; Springer Verlag, N.Y., 1987, pp. 87-133.

[27] Bakhtar, F.; So, K.S.: *A Study of Nucleating Flow of Steam in a Cascade of Supersonic Blading by the Time-Marching Method.* Int. J. Heat and Fluid Flow 12 (1991), pp. 54-62.

[28] Young, J.B.: *Two-Dimensional Non-Equilibrium Wet-Steam Calculations for Nozzles and Turbine Cascades.* Trans. ASME, J. Turbomachinery, 114 (1992), pp. 569-579.

[29] White, A.J.; Young, J.B.: *Time-Marching Method for the Prediction of Two-Dimensional, Unsteady Flows of Condensing Steam.* AIAA Journal Propulsion and Power, 9 (1993), pp. 579-587.

[30] Liberson, A.; Kosolapov, Y.; Rieger, N.; Hesler, S.: *Calculation of 3-D Condensing Flows in Nozzles and Turbine Stages.* EPRI Nucleation Workshop, Rochester, N.Y., October 24-26, 1995.

[31] Bakhtar, F.; Mahpeykar, M.R.; Abbas, K.K.: *An Investigation of Nucleating Flows of Steam in a Cascade of Turbine Blading - Theoretical Treatment.* Transaction of the ASME 117 (1995), pp. 138-144.

[32] White, A.J.; Young, J.B.; Walters, P.T.: *Experimental Validation of Condensing Flow Theory for a Stationary Cascade of Steam Turbine Blades.* Phil. Trans. R. Soc., London, A 354 (1996), pp. 59-88.

[33] Steger, J.L.: *Implicit Finite Difference Simulation of Flows About Two-Dimensional Arbitrary Geometries.* AIAA Journal, 17 (1978), pp. 679-686.

[34] Pulliam, T.H.; Steger, J.L.: *Implicit Finite Difference Simulations of Three-Dimensional Compressible Flows.* AIAA Journal, 18 (1980), pp. 159-167.

[35] Patankar, S.V.; Spalding, D.B.: *A Calculation Procedure for Heat, Mass and Momentum Transfer in Three-Dimensional Parabolic Flows.* Int. J. Heat Mass Transfer, 15 (1972), pp. 1787-1806.

[36] Aslan, A.R.; Grundmann, R.: *Computation of Three-Dimensional Subsonic Flows in Ducts Using the PNS Approach.* ZFW, 14 (1990), Springer Verlag, pp. 373-380.

[37] Kirtley, K.R.; Lakshminarayana, B.: *A Multiple Pass Space-Marching Method for Three-Dimensional Incompressible Viscous Flow.* ZFW, 16 (1992), Springer Verlag, pp. 49-59.

[38] Hollenbäck, D.M.; Blom, G.A.: *Application of a Parabolized Navier-Stokes Code to an HSCT Configuration and Comparison to Wind Tunnel Test Data.* AIAA Paper 93-3537, 1993.

[39] Krishnan, R.R.; Eidson, T.M.: *An Efficient, Parallel Space-Marching Euler Solver for HSCT Research.* AIAA Paper 95-1749, 1995.

[40] Nakahashi, K.; Saitoh, E.: *Space-Marching Method on Unstructured grid for Supersonic Flows with Embedded Subsonic Regions.* AIAA Paper 96-0418, 1996.

[41] Yamaleev, N.K.; Ballmann, J.: *Space-Marching Method for Calculating Steady Supersonic Flows on a Grid Adapted to the Solution.* J. Computational Physics, 146 (1998), pp. 436-463.

Chapter 3

Principles of Solution of the Governing Equations

In the previous chapter, we obtained the complete system of the Navier-Stokes/ Euler equations. We introduced additional thermodynamic relations for a perfect gas, and we also defined additional transport equations for a chemically reacting gas. Hence, we are now ready to solve the whole system of governing equations for the flow variables. As you can imagine, there exists a vast number of solution methodologies. If we do not consider analytical methods, which are applicable only to simplified flow problems, nearly all solution strategies follow the same path. First of all, the space, where the flow is to be computed – the *physical space*, is divided into a large number of geometrical elements called *grid cells*. This process is termed *grid generation* (some authors use the term *mesh* with identical meaning). It can also be viewed as placing first *grid points* (also called nodes or vertices) in the physical space and then connecting them by straight lines – *grid lines*. The grid normally consists in two dimensions of triangles or quadrilaterals, and in three dimensions of tetrahedra, hexahedra, prisms, or pyramids. The most important requirements placed on a grid generation tool are that there must not be any holes between the grid cells but also that the grid cells do not overlap. Additionally, the grid should be *smooth*, i.e., there should be no abrupt changes in the volume of the grid cells or in the stretching ratio and the elements should be as regular as possible. Furthermore, if the grid consists of quadrilaterals or hexahedra, there should be no large kinks in the grid lines. Otherwise, numerical errors would increase significantly.

On the one hand, the grid can be generated to follow closely the boundaries of the physical space, in which case we speak of *body-fitted* grid (Fig. 3.1a). The main advantage of this approach is that the flow can be resolved very accurately at the boundaries, which is essential in the case of shear layers along solid bodies. The price to be paid is a high degree of complexity of the grid generation tools, especially in the case of "real-life" geometries. On the other hand, the so-called *Cartesian* grids [1], [2], where the edges of the grid cells are oriented in parallel

to the Cartesian coordinates, can be generated very easily. Their advantage is that the evaluation of the fluxes in Eq. (2.19) is much more simple then for body-fitted grids. But, considering Fig. 3.1b it becomes clear that a general and accurate treatment of the boundaries is hard to achieve [3]. Because of this serious disadvantage, the body-fitted approach is preferred, particularly in the industrial environment, where the geometrical complexity of a configuration is usually very high.

Nowadays, the overwhelming number of numerical methods for the solution of the Euler- and the Navier-Stokes equations employ a separate discretisation in space and in time – the so-called *method of lines* [4]. Herewith, dependent on the particular algorithm chosen, the grid is used either to construct control volumes and to evaluate the flux integrals, or to approximate the spatial derivatives of the flow quantities. In a further step, the resulting time-dependent equations are advanced in time, starting from a known initial solution, with the aid of a suitable method. Another possibility, when the flow variables do not change in time, is to find the steady-state solution of the governing equations by means of an iterative process.

The way we derived the governing equations (2.19), the continuity equation (2.3) contains a time derivative of the density. Since the density, as an independent variable, is used to calculate the pressure (Eq. (2.29)), there is a coupling between the time evolution of the density and of the pressure in the momentum equations. Solution methods employing discretisations of the governing equations (2.19) are for this reason called *density-based* schemes. The problem with this formulation is that for an incompressible fluid the pressure is no longer driven by any independent variable, because the time derivative of the density disappears from the continuity equation. Another difficulty arises from the growing disparity between acoustic and convective wave speeds with decreasing Mach number, which makes the governing equations increasingly stiff and hence hard to solve [5]. Basically, three approaches were developed to cope with the problem. The first possibility is to solve a Poisson equation in pressure, which can be derived from the momentum equations [6]-[8]. These methods are denoted as *pressure-based*. The second approach, called the *artificial compressibility* method, is based on the idea to substitute time derivative of the pressure for that of the density in the continuity equation [9], [10]. In this way, velocity and pressure field are directly coupled. The third solution, and the most general one, is based on *preconditioning* of the governing equations [11]-[18]. This methodology allows it to employ the same numerical scheme for very low as well as high Mach number flows. We shall discuss this approach more extensively in Section 9.5.

In the following, we shall learn more about the very basic principles of various solution methodologies for the numerical approximation of the governing equations in space and in time, for the turbulence modelling, and also for the boundary treatment.

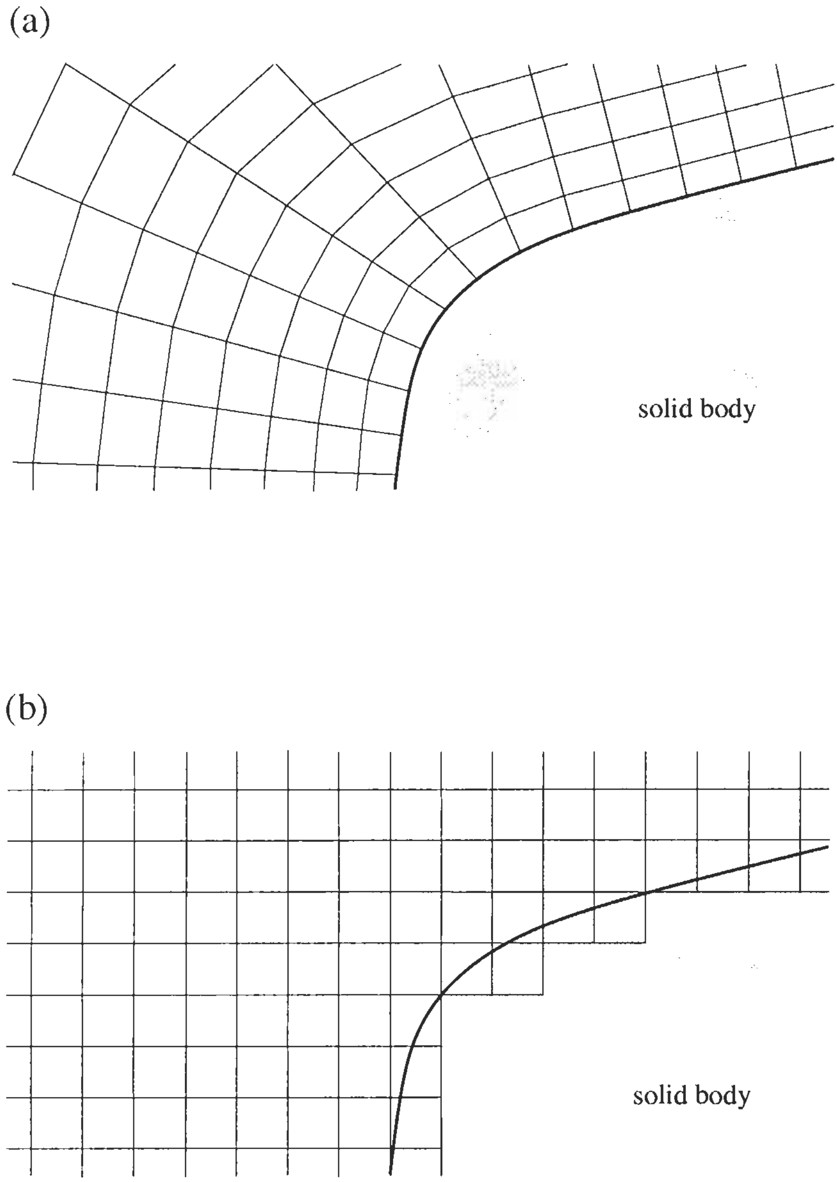


Figure 3.1: Body-fitted (a) and Cartesian grid (b) near a solid body (shown here in two dimensions).

3.1 Spatial Discretisation

Let us at the beginning turn our attention to the first step – the *spatial discretisation* of the Navier-Stokes equations, i.e., the numerical approximation of the convective and viscous fluxes, as well as of the source term. Many different methodologies were devised for this purpose in the past and the development still continues. In order to sort them, we can at first divide the spatial discretisation schemes into the following three main categories: *finite difference, finite volume*, and *finite element*. All these methods rely on some kind of grid in order to discretise the governing equations (2.19). There exist basically two types of grids:

- *Structured* grids (Fig. 3.2) – each grid point (vertex, node) is uniquely identified by the indices i, j, k and the corresponding Cartesian coordinates $x_{i,j,k}$, $y_{i,j,k}$, and $z_{i,j,k}$. The grid cells are quadrilaterals in 2-D and hexahedra in 3-D. If the grid is body-fitted, we also speak of *curvilinear* grid.

- *Unstructured* grids (Fig. 3.3) – grid cells as well as grid points have no particular ordering, i.e., neighbouring cells or grid points cannot be directly identified by their indices (e.g., cell 6 adjacent to cell 119). In the past, the grid cells were triangles in 2D and tetrahedra in 3D. Today, unstructured grids usually consist of a mix of quadrilaterals and triangles in 2D and of hexahedra, tetrahedra, prisms and pyramids in 3D, in order to resolve the boundary layers properly. Therefore, we speak in this case of *hybrid* or *mixed* grids.

The main advantage of structured grids follows from the property that the indices i, j, k represent a linear address space – also called the *computational space*, since it directly corresponds to how the flow variables are stored in the computer memory. This property allows it to access the neighbours of a grid point very quickly and easily, just by adding or subtracting an integer value to or from the corresponding index (e.g., like (i+1), (k-3), etc. – see Fig. 3.2). As one can imagine, the evaluation of gradients, fluxes, and also the treatment of boundary conditions is greatly simplified by this feature. The same holds for the implementation of an implicit scheme, because of the well-ordered, banded flux Jacobian matrix. But there is also a disadvantage. This is the generation of structured grids for complex geometries. As sketched in Fig. 3.4, one possibility is to divide the physical space into a number of topologically simpler parts – blocks – which can be more easily meshed. We therefore speak of *multiblock* approach [19]-[23]. Of course, the complexity of the flow solver is increased, since special logic is required to exchange physical quantities or fluxes between the blocks. Additional flexibility is added, if the grid points at both sides of an interface can be placed independently of each other, i.e., if the grid lines are allowed not to meet at a block boundary (like inside C or F in Fig. 3.4). Those grid points, which are located only on one side of a block interface are called *hanging nodes*. The advantage of this approach is quite obvious – the number

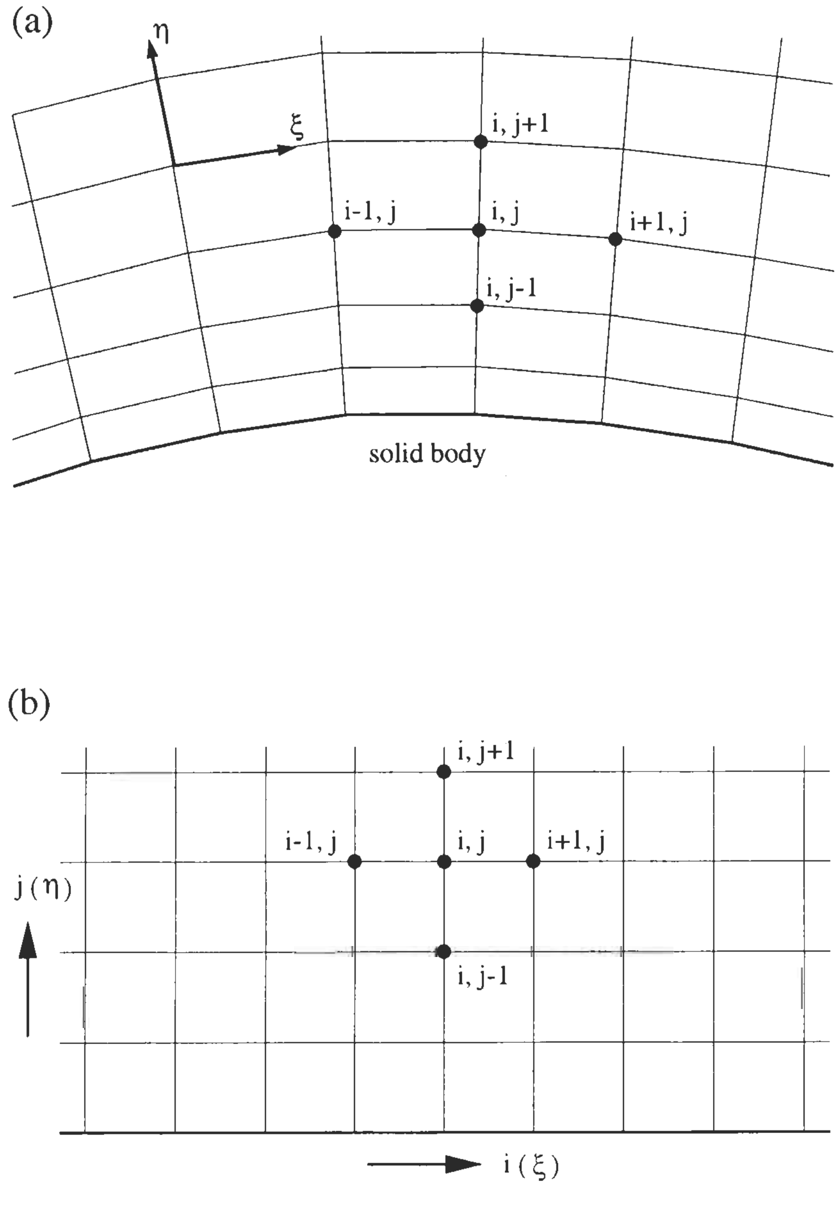


Figure 3.2: Structured, body-fitted grid approach (in two dimensions): (a) shows the physical space; (b) shows the computational space; ξ, η represent a curvilinear coordinate system.

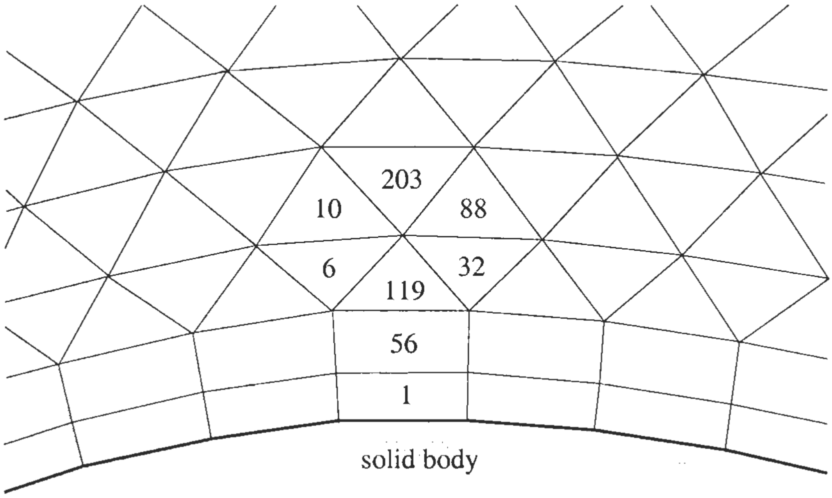


Figure 3.3: Unstructured, mixed grid approach; numbers mark individual cells.

of grid lines can be chosen separately for each block as required. The price paid for the enhanced flexibility is an increased overhead for the conservative treatment of the hanging nodes [24], [25]. The multiblock methodology also offers interesting possibilities with respect to the implementation of the flow solver on a parallel computer by means of *domain decomposition*. However, very long times (weeks or months) are still required for the grid generation in the case of complex configurations.

Another methodology, related to block structured grids, represents the so-called *Chimera technique* [26]-[30]. The basic idea here is to generate first the grids separately around each geometrical entity in the domain. After that, the grids are combined together in such a way that they overlap each other where they meet. The situation is depicted in Fig. 3.5 for a simple configuration. The crucial operation is an accurate transfer of quantities between the different grids at the overlapping region. Therefore, the extension of the overlap is adjusted accordingly to the required interpolation order. The advantage of the Chimera technique over the multiblock approach is the possibility to generate the particular grids completely independent of each other, without having to take care of the interface between the grids. On the other hand, the problem of the Chimera technique is that the conservation properties of the governing equations are not satisfied through the overlapping region.

The second type of grids are the unstructured grids. They offer the largest flexibility in the treatment of complex geometries [31]. The main advantage of the unstructured grids is based on the fact that triangular (2D) or tetrahedral (3D) grids can in principle be generated automatically, independent of the complexity of the domain. In practice, it is of course still necessary to set some parameters appropriately, in order to obtain a good quality grid. Furthermore,

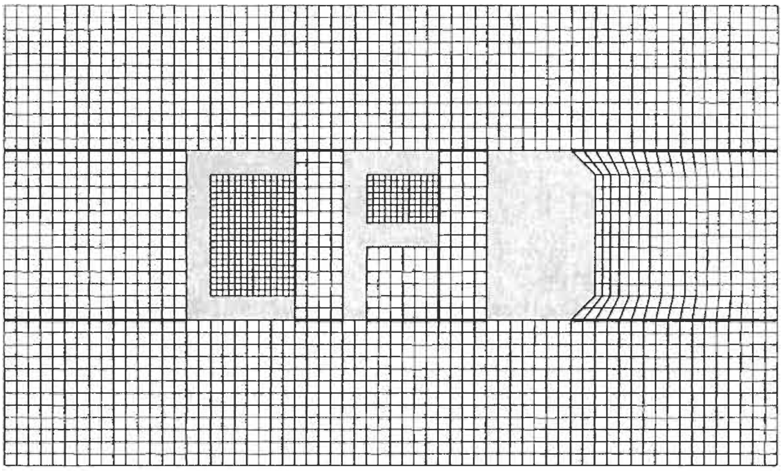


Figure 3.4: Structured, multiblock grid with conforming/non-conforming inter-block interfaces; thick lines represent block boundaries.

in order to resolve the boundary layers accurately, it is advisable to employ in 2D rectangular and in 3D prismatic or hexahedral elements near solid walls [32]-[39]. Another benefit of such mixed grids is the reduction of the number of grid cells, edges, faces and possibly also of grid points. But, one should keep in mind that the generation of mixed grids is non-trivial for geometrically demanding cases. However, the time required to construct an unstructured, mixed grid for a complex configuration is still significantly lower than the one required for a multiblock structured grid. Since nowadays, the geometrical fidelity of the flow simulations is rapidly increasing, the ability to generate grids fast and with minimum user interaction becomes more and more important. This is particularly true in industrial environment. Another advantage of the unstructured grids is that solution dependent grid refinement and coarsening can be handled in relatively native and seamless manner. To mention also the disadvantages of unstructured methods, one of them is the necessity to employ sophisticated data structures inside the flow solver. Such data structures work with indirect addressing, which, depending on the computer hardware, leads to more or less reduced computational efficiency. Also the memory requirements are in general higher as compared to the structured schemes. But despite all problems, the capability to handle complex flow problems in short turn-around times still weights much more. From this point, it is not surprising that for example nearly all vendors of commercial CFD software switched over to unstructured flow solvers. A detailed review of various methodologies for spatial and temporal discretisation on unstructured grids appeared recently in [40].

Having generated the grid, the next question is how to actually discretise the governing equations. As we already said, we can basically choose between three methodologies: finite differences, finite volumes, and finite elements. We

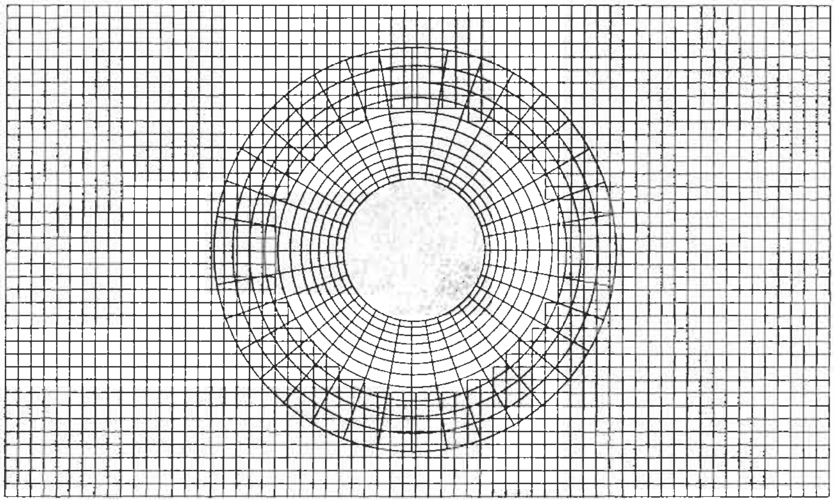


Figure 3.5: Illustration of the Chimera technique (2-D example).

want to discuss all of them briefly in the next subsections.

3.1.1 Finite Difference Method

The finite difference method was among the first methods applied to the numerical solution of differential equations. It was first utilised by Euler, probably in 1768. The finite difference method is directly applied to the differential form of the governing equations. The principle is to employ a Taylor series expansion for the discretisation of the derivatives of the flow variables. Let us for illustration consider the following example.

Suppose we would like to compute the first derivative of a scalar function $U(x)$ at some point x_0. If we develop now $U(x_0 + \Delta x)$ as a Taylor series in x, we obtain

$$U(x_0 + \Delta x) = U(x_0) + \Delta x \frac{\partial U}{\partial x}\bigg|_{x_0} + \frac{\Delta x^2}{2}\frac{\partial^2 U}{\partial x^2}\bigg|_{x_0} + \cdots \tag{3.1}$$

With this, the first derivative of U can be approximated as

$$\frac{\partial U}{\partial x}\bigg|_{x_0} = \frac{U(x_0 + \Delta x) - U(x_0)}{\Delta x} + \mathcal{O}(\Delta x)\,. \tag{3.2}$$

The above approximation is of first order, since the *truncation error* (abbreviated as $\mathcal{O}(\Delta x)$), which is proportional to the largest term of the remainder, goes to zero with the first power of Δx (for a discussion on the order of accuracy see Chapter 10). The same procedure can be applied to derive more accurate finite difference formulae and to obtain approximations to higher-order derivatives.

An important advantage of the finite difference methodology is its simplicity. Another advantage is the possibility to easily obtain high-order approximations, and hence to achieve high-order accuracy of the spatial discretisation. On the other hand, because the method requires a structured grid, the range of application is clearly restricted. Furthermore, the finite difference method cannot be directly applied in body-fitted (curvilinear) coordinates, but the governing equations have to be first transformed into a Cartesian coordinate system – or in other words – from the physical to the computational space (Fig. 3.2). The problem herewith is that the Jacobian of coordinate transformation appears in the flow equations (see, e.g., Appendix A.1). This Jacobian has to be consistently discretised in order to avoid the introduction of additional numerical errors. Thus, the finite difference method can be applied only to rather simple geometries. Nowadays, it is sometimes utilised for the direct numerical simulation of turbulence (DNS), but it is only very rarely used for industrial applications. More details to the finite difference method can be found for example in [41], or in textbooks on the solution of partial differential equations.

3.1.2 Finite Volume Method

The finite volume method directly utilises the conservation laws – the integral formulation of the Navier-Stokes/Euler equations. It was first employed by McDonald [42] for the simulation of 2-D inviscid flows. The finite volume method discretises the governing equations by first dividing the physical space into a number of arbitrary polyhedral control volumes. The surface integral on the right-hand side of Equation (2.19) is then approximated by the sum of the fluxes crossing the individual faces of the control volume. The accuracy of the spatial discretisation depends on the particular scheme with which the fluxes are evaluated.

There are several possibilities of defining the shape and position of the control volume with respect to the grid. Two basic approaches can be distinguished:

- *Cell-centred* scheme (Fig. 3.6a) – here the flow quantities are stored at the centroids of the grid cells. Thus, the control volumes are identical to the grid cells.
- *Cell-vertex* scheme (Fig. 3.6b) – here the flow variables are stored at the grid points. The control volume can then either be the union of all cells sharing the grid point, or some volume centred around the grid point. In the former case we speak of *overlapping* control volumes, in the second case of *dual* control volumes.

We shall discuss the advantages and disadvantages of cell-centred and cell-vertex formulations in both chapters on spatial discretisation.

The main advantage of the finite volume method is that the spatial discretisation is carried out directly in the physical space. Thus, there are no problems with any transformation between coordinate systems, like in the case of the finite difference method. Compared to the finite differences, one further advantage

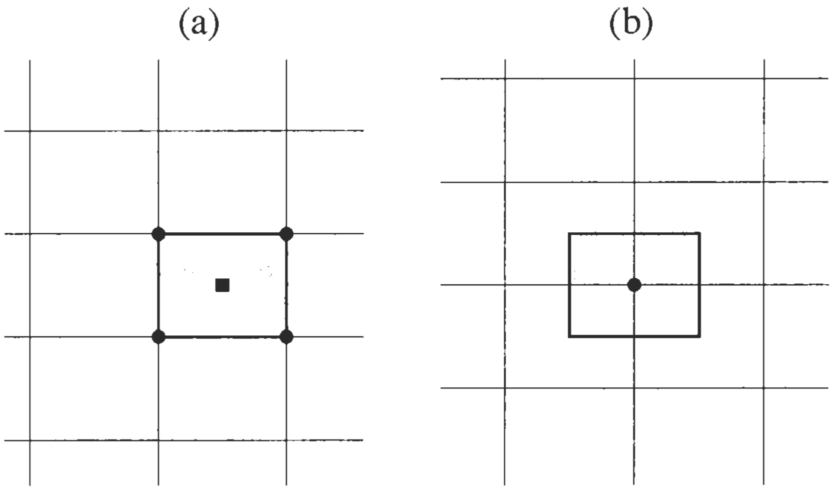


Figure 3.6: Control volume of cell-centred (a) and cell-vertex (b) scheme (dual control volume).

of the finite volume method is that it is very flexible – it can be rather easily implemented on structured as well as on unstructured grids. This renders the finite volume method particularly suitable for the treatment of flows in complex geometries.

Since the finite volume method is based on the direct discretisation of the conservation laws, mass, momentum and energy are also conserved by the numerical scheme. This leads to another important feature of the method, namely the ability to compute *weak solutions* of the governing equations correctly. However, in the case of the Euler equations, one additional condition has to be fulfilled. This is known as the *entropy condition.* It is necessary because of the non-uniqueness of the weak solutions. The entropy condition prevents the occurrence of unphysical features like expansion shocks, which violate the second law of thermodynamics (decrease the entropy). As a further consequence of the conservative discretisation, the Rankine-Hugoniot relations, which must hold across a solution discontinuity (such as a shockwave or a contact discontinuity), are satisfied directly.

It is interesting to note that under certain conditions, the finite volume method can be shown to be equivalent to the finite difference method, or to a low-order finite element method. Because of its attractive properties, the finite volume method is nowadays very popular and in wide use. It will be presented in the following chapters.

3.1.3 Finite Element Method

The finite element method was originally employed for structural analysis only. It was first introduced by Turner et al. [43] in 1956. About ten years later, researchers started to use the finite element method also for the numerical solution of field equations in continuous media. However, only with the beginning of the 90's, did the finite element method gain popularity in the solution of the Euler and the Navier-Stokes equations. A good introduction into the classical finite element methodology can be found in [44]. Applications to flow problems are described in [45], [46], and more recently in [47].

The finite element method, as it is in general applied to the solution of the Euler/Navier-Stokes equations, starts with a subdivision of the physical space into triangular (in 2-D) or into tetrahedral (in 3-D) elements. Thus, an unstructured grid has to be generated. Depending on the element type and the required accuracy, a certain number of points at the boundaries and/or inside an element is specified, where the solution of the flow problem has to be found. The total number of points multiplied with the number of unknowns determines the number of *degrees of freedom*. Furthermore, the so-called *shape functions* have to be defined, which represent the variation of the solution inside an element. In practical implementations, linear elements are usually employed, which use the grid nodes exclusively. The shape functions are then linear distributions, whose value is zero outside the corresponding element. This results in a second-order accurate representation of the solution on smooth grids.

Within the finite element method, it is necessary to transform the governing equations from the differential into an equivalent integral form. This can be accomplished in two different ways. The first one is based on the *variational principle*, i.e., a physical solution is sought, for which a certain functional possesses an extremum. The second possibility is known as the *method of weighted residuals* or the *weak formulation*. Here, it is required that the weighted average of the residuals is identically zero over the physical domain. The residuals can be viewed as the errors of the approximation of the solution. The weak formulation has the same advantage as the finite volume discretisation of the conservation laws – it allows the treatment of discontinuous solutions such as shocks. Therefore, the weak formulation is preferred over the variational methodology.

The finite element method is attractive because of its integral formulation and the use of unstructured grids, which are both preferable for flows in or around complex geometries. The method is also particularly suitable for the treatment of non-Newtonian fluids. The finite element method has a very rigorous mathematical foundation, particularly for elliptic and parabolic problems. Although it can be shown in certain cases that the method is mathematically equivalent to the finite volume discretisation, the numerical effort is much higher. This may explain why the finite volume method became more popular. However, both methods are sometimes combined – particularly on unstructured grids. So for example, the treatment of the boundaries and the discretisation of the viscous fluxes is usually "borrowed" from the finite element method.

3.1.4 Other Discretisation Methods

There are few other numerical schemes which are in general not used in practice, but which are, in certain situations, superior to the methods discussed above. Two particular approaches should be mentioned here briefly.

Spectral Element Method

The first is for example the *Spectral Element Method* [48]-[51]. The spectral element method combines the geometric flexibility of the finite element technique with the high-order spatial accuracy (e.g., 10th-order) and the rapid convergence rate of the spectral schemes [52]. The method is based on a high-order polynomial representation of the solution (usually Lagrangian interpolants), combined with a standard Galerkin finite element method, or the method of weighted residuals. The spectral element method is appropriate either for a particular problem in which high-order regularity is guaranteed, or for which high-order regularity is not the exception, like in incompressible fluid mechanics. It is especially suitable for vortical flows. The advantage of the spectral element method is primarily its non-diffusive, non-dispersive approximation of the convection operator, and its good approximation of convection-diffusion boundary layers. The method can treat geometrically and physically complex problems, supposed the condition of high-order regularity is fulfilled. Apart from the rather narrow range of applications, the principal disadvantage of the spectral element method is its very high numerical effort as compared for example to the finite volume method.

Gridless Method

Another discretisation scheme, which gained recently some interest, is the so-called *Gridless Method* [53], [54]. This method employs only clouds of points for the spatial discretisation. It does not require that the points are connected to form a grid as in conventional structured or unstructured grid schemes. The gridless method is based on the differential form of the governing equations, written in the Cartesian coordinate system. Gradients of the flow variables are determined by a least-squares reconstruction, using a specified number of neighbours surrounding the particular point. The gridless method is neither a finite difference nor a finite volume or a finite element approach since coordinate transformations, face areas or volumes do not have to be computed. It can be viewed as a mix between the finite difference and the finite element method. The principal advantages of the gridless method are its flexibility in solving flows about complex configurations (similar to unstructured methods), and the possibility to locate or cluster the points (or the clouds of points) where it is appropriate. For example, it would be easily possible to select only the neighbours in the characteristic directions when computing gradients. However, there is one unresolved problem. Although the gridless method solves the conservation law form of the Euler or the Navier-Stokes equations, it is not clear whether conservation of mass, momentum and energy is really ensured.

Whichever spatial discretisation scheme we might select, it is important to ensure that the scheme is consistent, i.e., that it converges to the solution of the **discretised** equations, when the grid is sufficiently refined. It is therefore very important to check how much the solution changes, if the grid is refined (e.g., if we would double the number of grid points). If the solution improves only marginally, we speak of *grid converged* solution. Another rather self-consistency requirement is that the discretisation scheme possesses the order of accuracy, which is appropriate for the flow problem being solved. This rule is sometimes given up in favour of faster convergence, particularly in industrial environment (bad solution is better than no solution). This is of course a very dangerous practice. We shall return to the question of accuracy, stability and consistency later in Chapter 10.

3.1.5 Central versus Upwind Schemes

So far, we discussed only the basic choices which exist for the spatial discretisation. But within each of the above three main methods – finite difference, finite volume and finite element – various numerical schemes exist to perform the spatial discretisation. In this context, it is convenient to differentiate between the discretisation of the convective and the viscous fluxes ($\vec{F}_c$ and $\vec{F}_v$ in Eq. (2.19), respectively). Because of the physical nature of the viscous fluxes, the only reasonable way is to employ central differences (central averaging) for their discretisation. Thus, their discretisation on structured grids is straightforward. On unstructured triangular or tetrahedral grids, the viscous fluxes are best approximated using the Galerkin finite element methodology, even in the case of a finite volume scheme [55]. The situation becomes more complicated for unstructured mixed grids, where a modified averaging of gradients is more appropriate [56]-[60].

However, the real variety is found in the discretisation of the convective fluxes. In order to classify the individual methodologies, we will restrict our attention to schemes developed for the finite volume method, although most of the concepts are also directly applicable to the finite difference or the finite element method.

Central Schemes

To the first category we may count schemes, which are based solely on central difference formulae or on central averaging, respectively. Those are denoted as *central schemes*. The principle is to average the conservative variables to the left and to the right in order to evaluate the flux at a side of a control volume. Since the central schemes cannot recognise and suppress an odd-even decoupling of the solution (i.e., the generation of two independent solutions of the **discretised** equations), the so-called *artificial dissipation* (because of its similarity to the viscous terms) has to be added for stabilisation. The most widely known implementation is due to Jameson et al. [61]. On structured grids, it is based on a blend of 2nd- and 4th-differences scaled by the maximum eigenvalue of the con-

vective flux Jacobian. A combination of an undivided Laplacian and biharmonic operator is employed on unstructured grids [62]. The scheme can be improved remarkably using different scaling factors for each equation. This approach is known as the *matrix dissipation* scheme [63]. It should be mentioned that on unstructured, mixed element grids the explicit Runge-Kutta time-stepping scheme can become unstable, when combined with the conventional central scheme [64].

Upwind Schemes

On the other hand, there are more advanced spatial discretisation schemes, which are constructed by considering the physical properties of the Euler equations. Because they distinguish between upstream and downstream influences (wave propagation directions), they are termed *upwind schemes*. They can be roughly divided into four main groups:

- *flux-vector splitting*,
- *flux-difference splitting*,
- *total variation diminishing* (TVD), and
- *fluctuation-splitting* schemes.

Each of these is described briefly in the following.

Flux-Vector Splitting Schemes

One class of the flux-vector splitting schemes decomposes the vector of the convective fluxes into two parts according to the sign of certain characteristic variables, which are in general similar to but not identical with the eigenvalues of the convective flux Jacobian. The two parts of the flux vector are then discretised by upwind biased differences. The very first flux-vector splitting schemes of this type were developed in the beginning of the 1980's by Steger and Warming [65] and by Van Leer [66], respectively. A second class of flux-vector splitting schemes decompose the flux vector into a convective and a pressure (an acoustic) part. This idea is utilised by schemes like AUSM (*Advection Upstream Splitting Method*) of Liou et al. [67], [68], or the CUSP scheme (*Convective Upwind Split Pressure*) of Jameson [69], [70], respectively. Further similar approaches are the *Low-Diffusion Flux-Splitting Scheme* (LDFSS) introduced by Edwards [71], or the *Mach number-based Advection Pressure Splitting (MAPS)* scheme of Rossow [72], [73]. The second group of flux-vector splitting schemes gained recently larger popularity particularly because of their improved resolution of shear layers, but only a moderate computational effort. An advantage of the flux-vector splitting schemes is also that they can be quite easily extended to real gas flows, as opposed to flux-difference splitting or TVD schemes. We shall return to real gas simulations further below.

Flux-Difference Splitting Schemes

The second group – flux-difference splitting schemes – is based on the solution of the locally one-dimensional Euler equations for discontinuous states at an interface. This corresponds to the Riemann (shock tube) problem. The values on either side of the interface are generally termed as the *left* and *right* state. The idea to solve the Riemann problem at the interface between two control volumes was first introduced by Godunov [74] back in 1959. In order to reduce the numerical effort required for an exact solution of the Riemann problem, *approximate* Riemann solvers were developed, e.g., by Osher et al. [75] and Roe [76]. Roe's solver is often used today because of its excellent resolution of boundary layers and a crisp representation of shocks. It can be easily implemented on structured as well as on unstructured grids [77].

TVD Schemes

The idea of TVD schemes was first introduced by Harten [78] in 1983. The TVD schemes are based on a concept aimed at preventing the generation of new extrema in the flow solution. The principal conditions for a TVD scheme are that maxima must be non-increasing, minima non-decreasing, and no new local extrema may be created. Such a scheme is called *monotonicity preserving*. Thus, a discretisation methodology with TVD properties allows it to resolve a shock wave without any spurious oscillations of the solution. The TVD schemes are in general implemented as an average of the convective fluxes combined with an additional dissipation term. The dissipation term can either depend on the sign of the characteristic speeds or not. In the first case, we speak of an *upwind* TVD scheme [79], in the second case of a *symmetric* TVD scheme [80]. The experience shows that the upwind TVD scheme should be preferred since it offers a better shock and boundary layer resolution than the symmetric TVD scheme. The disadvantage of the TVD schemes is that they cannot be easily extended to higher than second-order spatial accuracy. This limitation can be overcome using the ENO (*Essentially Non-Oscillatory*) discretisation schemes [81]-[86].

Fluctuation-Splitting Schemes

The last group – the fluctuation-splitting schemes – provides for true multidimensional upwinding. The aim is to resolve accurately also those flow features which are not aligned with the grid. This is a significant advantage over all above upwind schemes, which split the equations according only to the orientation of the grid cells. Within the fluctuation-splitting methodology, the flow variables are associated with the grid nodes. Intermediate residuals are computed as flux balances over the grid cells, which consists of triangles in 2D and of tetrahedra in 3D. The cell-based residuals are then distributed in an upwind-biased manner to the nodes. After that, the solution is updated using the nodal values. In the case of systems of equations (Euler or Navier-Stokes), the cell-based resid-

uals have to be decomposed into scalar waves. Since the decomposition is not unique in 2D and in 3D, several approaches were developed in the past. The variety reaches from the wave model of Roe [87], [88] over the algebraic scheme of Sidilkover [89] to the most advanced characteristic decomposition method [90]-[93]. Despite the above mentioned advantage over the dimensionally split Riemann, TVD, etc. solvers, the fluctuation-splitting are so far used only in research codes. This can be attributed to the complexity and the high numerical effort, as well as to convergence problems.

Central versus Upwind Schemes

You may now ask, what are the benefits and the drawbacks of the individual spatial discretisation methods. Generally speaking, central schemes require lower numerical effort, and hence less CPU time per evaluation, compared to upwind schemes. On the other hand, upwind schemes are able to capture discontinuities much more accurately than central schemes. Furthermore, because of their lower numerical diffusion, the upwind schemes can resolve boundary layers using less grid points. Particularly, Roe's flux-difference splitting scheme allows a very accurate computation of boundary layers. The negative side of the upwind schemes emerges for second- or higher-order spatial accuracy. The problem is that the so-called *limiter functions* (or simply *limiters*) have to be employed in order to prevent the generation of spurious oscillations near strong discontinuities. Limiters are known to stall the convergence of an iteration scheme, because of accidental switching in smooth flow regions. A remedy was suggested by Venkatakrishnan [94]-[96], which works satisfactorily for most practical cases. But, small wiggles in the solution must be taken into account. Another disadvantage of the limiter functions is that they require high computational effort, particularly on unstructured grids.

Upwind Schemes for Real Gas Flows

With respect to real gas simulations, and in particular to chemically reacting flows, several extensions of the upwind discretisation schemes were presented. For the case of fluids in thermodynamic and chemical equilibrium, modifications of the Van Leer flux-vector splitting [66] and of the Roe approximate Riemann solver [76] were described in [97]-[99]. Formulations of the both upwind methods for the more complex case of flows with non-equilibrium chemistry and thermodynamics were provided in [100]-[105] and in the references cited therein. In particular, the articles [99], [103] and [104], respectively, give a good overview of the methodologies employed for the upwind discretisation schemes. Recently, a summary of the governing equations together with Jacobian and transformation matrices, which may be required by an upwind scheme, was presented in [106] for chemically reacting flows.

3.2 Temporal Discretisation

As already mentioned at the beginning of this chapter, the prevailing number of numerical schemes for the solution of the Euler and the Navier-Stokes equations applies the method of lines, i.e., a separate discretisation in space and in time. This approach offers the largest flexibility, since different levels of approximation can be easily selected for the convective and the viscous fluxes, as well as for the time integration – just as required by the problem solved. Therefore, we shall follow this methodology here. For the discussion of other methods, where time and space discretisation is coupled, the reader is referred to [41].

When the method of lines is applied to the governing equations (2.19), it leads, written down for each control volume, to a system of coupled ordinary differential equations in time

$$\frac{d(\Omega \bar{M} \vec{W})}{dt} = -\vec{R}. \tag{3.3}$$

For clarity, we omitted any cell indices. In Eq. (3.3), Ω denotes volume of the control volume and $\vec{R}$ stands for the complete spatial (finite volume) discretisation including the source term – the so-called *residual*. The residual is a non-linear function of the conservative variables $\vec{W}$. Finally, $\bar{M}$ represents what is termed the *mass matrix*. For a cell-vertex scheme, it relates the average value of $\vec{W}$ in the control volume to the point values at the associated interior node and the neighbouring nodes [107], [108]. In the case of a cell-centred scheme, the mass matrix can be substituted by an identity matrix, without compromising the temporal accuracy of the scheme. The same holds for a cell-vertex scheme applied on a **uniform** grid, since then the nodes coincide with the centroids of the control volumes. The mass matrix is only a function of the grid and couples the system of differential equations (3.3). For steady-state cases, where time accuracy is not a concern, the mass matrix can be "lumped", i.e., replaced by the identity matrix. In this way, the expensive inversion of $\bar{M}$ can be avoided and the system (3.3) is decoupled. In this respect, it is important to realise that at the steady-state, the solution accuracy is determined solely by the approximation order of the residual. Thus, the mass matrix becomes only important for cell-vertex schemes in unsteady calculations.

If we assume a static grid, we may take the volume Ω and the mass matrix outside the time derivative. Then, we can approximate the time derivative by the following non-linear scheme [41]

$$\frac{\Omega \bar{M}}{\Delta t} \Delta \vec{W}^n = -\frac{\beta}{1+\omega} \vec{R}^{n+1} - \frac{1-\beta}{1+\omega} \vec{R}^n + \frac{\omega \Omega \bar{M}}{(1+\omega)\Delta t} \Delta \vec{W}^{n-1} \tag{3.4}$$

with

$$\Delta \vec{W}^n = \vec{W}^{n+1} - \vec{W}^n \tag{3.5}$$

being the solution correction. The superscripts n and $(n+1)$ denote the time levels (n means the current one). Furthermore, Δt represents the time step.

The scheme in Eq. (3.4) is 2nd-order accurate in time if the condition

$$\beta = \omega + \frac{1}{2} \tag{3.6}$$

is fulfilled, otherwise the time accuracy is reduced to 1st-order. Depending on the settings of the parameters β and ω, we can obtain either *explicit* ($\beta = 0$) or *implicit* time-stepping schemes. We shall discuss this two main classes briefly in the following paragraphs, and in more detail later in Chapter 6.

3.2.1 Explicit Schemes

A basic explicit time-integration scheme is obtained by setting $\beta = 0$ and $\omega = 0$ in Eq. (3.4). In this case, the time derivative is approximated by a forward difference and the residual is evaluated at the current time level only (based on known flow quantities), i.e.,

$$\Delta \vec{W}^n = -\frac{\Delta t}{\Omega} \vec{R}^n \,, \tag{3.7}$$

where the mass matrix was lumped. This represents a *single-stage* scheme, because a new solution $\vec{W}^{n+1}$ results from only one evaluation of the residual. The scheme Eq. (3.7) is of no practical value, since it is stable only if combined with a first-order upwind spatial discretisation.

Very popular are *multistage* time-stepping schemes (Runge-Kutta schemes), where the solution is advanced in several stages [61] and the residual is evaluated at intermediate states. Coefficients are used to weight the residual at each stage. The coefficients can be optimised in order to expand the stability region and to improve the damping properties of the scheme and hence its convergence and robustness [61], [109], [110]. Also, depending on the stage coefficients and the number of stages, a multistage scheme can be extended to 2nd- or higher-order accuracy in time. Special Runge-Kutta schemes were also designed to preserve the properties of the TVD and ENO spatial discretisation methods, while maximising the allowable time step [111].

Explicit multistage time-stepping schemes can be employed in connection with any spatial discretisation scheme. They can be easily implemented on serial, vector, as well as on parallel computers. Explicit schemes are numerically cheap, and they require only a small amount of computer memory. On the other hand, the maximum permissible time step is severely restricted because of stability limitations. Particularly for viscous flows and highly stretched grid cells, the convergence to steady state slows down considerably. Furthermore, in the case of stiff equation systems (e.g., real gas simulation, turbulence models), or stiff source terms, it can take extremely long to achieve the steady state. Or even worse, an explicit scheme may become unstable or lead to spurious steady solutions [112].

If we are interested in steady-state solutions only, we can select from (or combine) several convergence acceleration methodologies. The first, and very common, technique is *local time-stepping*. The idea is to advance the solution

in each control volume with the maximum permissible time step. As a result, the convergence to the steady state is considerably accelerated. However, the transient solutions are no longer temporally accurate. Another approach is the so-called *characteristic time-stepping.* Here, not only locally varying time steps are used, but also each equation (continuity, momentum and energy equations) is integrated with its own time step. The potential of this concept was presented for 2-D Euler equations in Ref. [113]. A further acceleration technique, which is similar to the characteristic time-stepping is *Jacobi preconditioning* [114], [115], [116]. It is basically a point-implicit Jacobi relaxation, which is carried out at each stage of a Runge-Kutta scheme. Jacobi preconditioning can be seen as a time-stepping in which all wave components (eigenvalues of the flux Jacobian) are scaled to have the same effective speed. It also adds an implicit component to the basic explicit scheme.

An other very popular acceleration method is aimed at increasing the maximum possible time step by introducing a certain amount of implicitness in the explicit scheme. It is termed *implicit residual smoothing* or *residual averaging* [117], [118]. On a structured grid, the method requires the solution of a tridiagonal matrix for each conservative variable. In the case of unstructured grids, the matrix is usually inverted by means of Jacobi iteration. The standard implicit residual smoothing allows an increase of the time step by factor of 2-3. Several other implicit residual smoothing techniques were developed. For example the *upwind implicit residual smoothing* methodology [119], which was designed to be employed together with an upwind spatial discretisation. In comparison to the standard technique, it allows for significantly larger time steps and it also improves the robustness of the time-stepping process [120]. One further method is the *implicit-explicit residual smoothing* [121], [122], which is intended to improve the damping properties of the time discretisation at larger time steps.

The last and probably the most important convergence acceleration technique, which should be mentioned here, is the *multigrid method.* It was developed in the 1960's in Russia by Fedorenko [123] and Bakhvalov [124]. They applied multigrid for the solution of elliptic boundary-value problems. The methodology was further developed and promoted by Brandt [125], [126]. The idea of multigrid is based on the observation that iterative schemes usually eliminate high-frequency errors in the solution (i.e., oscillations between the grid nodes) very effectively. On the other hand, they perform quite poor in reducing low-frequency (i.e., global) solution errors. Therefore, after advancing the solution on a given grid, it is transferred to a coarser grid, where the low-frequency errors become partly high-frequency ones and where they are again effectively damped by an iterative solver. The procedure is repeated recursively on a sequence of progressively coarser grids, where each *multigrid level* helps to annihilate a certain bandwidth of error frequencies. After the coarsest grid is reached, the solution corrections are successively collected and interpolated back to the initial fine grid, where the solution is then updated. This complete *multigrid cycle* is repeated until the solution changes less than a given threshold. In order to accelerate the convergence even further, it is possible to start the multigrid process on a coarse grid, carry out a number of cycles

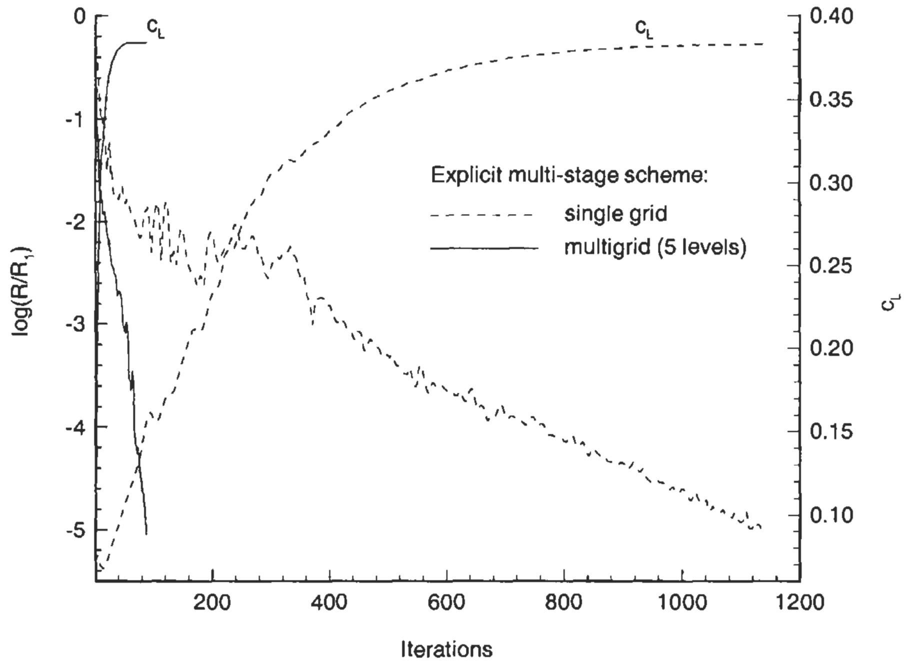


Figure 3.7: Convergence history for inviscid transonic flow past NACA 0012 airfoil; R = density residual, c_L = lift coefficient.

and then to transfer the solution to a finer grid, where the multigrid cycles are performed again. The procedure is then successively repeated until the finest grid is reached. This methodology is known as *Full Multigrid* (FMG) [126].

As already mentioned, the multigrid method was originally developed for the solution of elliptic boundary-value problems (Poisson equation), where it is very efficient. Jameson proposed first to employ multigrid also for the solution of the Euler equations [117], [127]. The approach was based on the so-called *Full Approximation Storage* (FAS) scheme [126], where multigrid is directly applied to the non-linear governing equations. Nowadays, multigrid represents a standard acceleration technique for the solution of the Navier-Stokes equations. Examples of implementations can be found in Refs. [128]-[133] for structured grids, and in Refs. [134]-[143] for unstructured grids. Although not as fast as in the case of elliptic differential equations, it was often demonstrated that multigrid can accelerate the solution of the Euler or the Navier-Stokes equations by a factor between 5 and 10. An example for transonic flow is shown in Fig. 3.7. Recent research also revealed that faster convergence can be achieved if the governing equations are decomposed into hyperbolic and elliptic parts [144]. We shall return to the multigrid methodology again in Section 9.4.

3.2.2 Implicit Schemes

A family of implicit time integration schemes is obtained from Eq. (3.4) by setting $\beta \neq 0$. Very popular for the simulation of unsteady flows is the 3-point implicit backward-difference scheme with $\beta = 1$ and $\omega = 1/2$, which is 2nd-order accurate in time. In this case, the scheme is mostly employed within the so-called *dual time-stepping* approach [145]-[147], [107], [108], where a steady-state problem is solved in pseudo-time at each physical time step.

For the solution of stationary flow problems, a scheme with $\omega = 0$ is more suitable, since it requires less computer storage. Herewith, if we linearise the residual $\vec{R}^{n+1}$ in Eq. (3.4) about the current time level, we obtain the scheme

$$\left(\bar{M} \frac{\Omega}{\Delta t} + \beta \frac{\partial \vec{R}}{\partial \vec{W}} \right) \Delta \vec{W}^n = -\vec{R}^n . \tag{3.8}$$

The term $\partial \vec{R} / \partial \vec{W}$ is denoted as the flux Jacobian. It constitutes a large sparse matrix. The expression enclosed in parenthesis on the left-hand side of Eq. (3.8) is also referred to as the *implicit operator*. As already discussed above, the mass matrix $\bar{M}$ can be replaced by the identity matrix, without influencing the steady state solution. The parameter β in Eq. (3.8) is generally set to 1, which results in a 1st-order accurate temporal discretisation. A 2nd-order time accurate scheme is obtained for $\beta = 1/2$. However, this is not advised since the scheme with $\beta = 1$ is much more robust, and the time accuracy plays no role for steady problems anyway.

The principal advantage of implicit schemes as compared to explicit ones is that significantly larger time steps can be used, without hampering the stability of the time integration process. In fact, for $\Delta t \to \infty$ the scheme (3.8) transforms into standard Newton's method, which allows for quadratic convergence. However, the condition for quadratic convergence is that the flux Jacobian contains the complete linearisation of the residual. Another important advantage of implicit schemes is their superior robustness and convergence speed in the case of stiff equation systems and/or source terms, which are often encountered in real gas simulations, turbulence modelling, or in the case of highly stretched grids (high Reynolds number flows). On the other hand, the faster (in terms of time steps or iterations) and the more robust an implicit scheme is, the higher is usually the computational effort per time step or iteration. Therefore, an explicit scheme accelerated by multigrid can be equally or even more efficient. Furthermore, implicit schemes are significantly more difficult to vectorise or to parallelise than their explicit counterparts.

The implicit scheme (3.8), written for each grid point, represents a large system of linear equations, which has to be solved for the update $\Delta \vec{W}^n$ at each time step Δt. This task can be accomplished using either a *direct* or an *iterative* method.

The direct methods are based on the exact inversion of the left-hand side of Eq. (3.8) using either the Gaussian elimination or some direct sparse matrix method [148], [149]. Although quadratic convergence was demonstrated

on structured [150]-[153] as well as on unstructured grids [154], direct methods are not an option for 3-D problems because they require an excessively high computational effort and a huge amount of computer memory.

Thus, the only practical method for larger grids or 3-D problems are iterative methods. Here, the linear system is solved for $\Delta\vec{W}^n$ at each time step using some iterative matrix inversion methodology. In order to reduce the memory requirements and also to increase the diagonal dominance, the flux Jacobian $\partial\vec{R}/\partial\vec{W}$ is mostly based on linearisation of a 1st-order accurate spatial discretisation of the right-hand side. The two main consequences of this approximation are that the quadratic convergence of Newton's scheme cannot be reached and that the maximum time step becomes limited. On the other hand, the numerical effort of an iteration step is significantly reduced, which leads to a numerically highly efficient scheme.

In the case of structured grids, iterative methods like the *Alternating Direction Implicit* (ADI) scheme [155]-[158], the (line) Jacobi or the Gauss-Seidel relaxation scheme [159]-[163], and particularly the *Lower-Upper Symmetric Gauss-Seidel* (LU-SGS; also referenced to as LU-SSOR – Lower-Upper Symmetric Successive Overrelaxation) scheme [164]-[168] are mainly employed. All these methods are based on splitting of the implicit operator into a sum or product of parts, which can be each more easily inverted. Because of the associated *factorisation error* and also the simplification of the flux Jacobian, it does not pay off to solve the linear system very accurately. In fact, only one iteration is carried out at each time step of the ADI and the LU-SGS method.

Implicit iterative methods for unstructured grids are in the most cases based on the Gauss-Seidel relaxation scheme [169]-[172]. In order to improve the convergence, it is possible to use the red-black Gauss-Seidel methodology. Its extension to unstructured grids was demonstrated in [173]-[175]. A particularly interesting possibility is also offered by an implementation of the LU-SGS scheme on unstructured grids [18], [176], [177], because of its very low memory requirements and numerical effort.

Because of the success of the line-implicit methods on structured grids, a few attempts were made to adopt this methodology on unstructured grids [178], [179]. The approach was to construct continuous lines such that each grid point or each grid cell (in the case of a cell-centred scheme) is visited only once – a so-called Hamiltonian tour [180]. The lines were oriented primarily in coordinate directions, but they were folded at the boundaries and where necessary (therefore they were nicknamed "snakes"). A tri-diagonal solver was then employed to invert the left-hand side of Eq. (3.8). Later on, it was recognised that folding the lines can slow down the convergence. To overcome this, each line was broken into multiple *linelets* [181]. However, the performance on a vector computer was rather poor. The idea of linelets was also employed to improve the convergence of an explicit scheme on highly stretched viscous unstructured grids using an implicit solver in the direction across the boundary layer [142].

More sophisticated iterative techniques, which treat the linear equation system in a more global way, are the so-called *Krylov subspace* methods. Their development was triggered by the introduction of an efficient iterative scheme

for solving large, sparse linear systems – namely the ***conjugate gradient*** method [182] by Hestenes and Stiefel. The original conjugate gradient method is restricted to Hermitian positive definite matrices only, but for an $n \times n$ matrix it converges in at most n iterations. Since then, a variety of Krylov subspace methods was proposed for the solution of arbitrary non-singular matrices, as they occur in CFD applications. For example, there are methods like CGS (*Conjugate Gradient Squared*) [183], Bi-CGSTAB (*Bi-Conjugate Gradient Stabilised*) [184], or TFQMR (*Transpose-Free Quasi-Minimum Residual*) [185].

However, the most widely employed method is GMRES (*Generalised Minimal Residual*) developed by Saad and Schultz [186]. If we rewrite the implicit scheme (Eq. (3.8)) as

$$\bar{J}\,\Delta\vec{W}^n = -\vec{R}^n\,, \tag{3.9}$$

then $\bar{J}$ represents a large, sparse, and non-symmetric matrix (the left-hand side). Starting from an initial guess $\Delta\vec{W}_0$, the GMRES(m) method seeks a solution $\Delta\vec{W}^n$ in the form $\Delta\vec{W}^n = \Delta\vec{W}_0^n + \vec{y}_m$, where $\vec{y}_m$ belongs to the Krylov subspace

$$\begin{aligned} \mathcal{K}_m &\equiv \text{span}\{\vec{r}_0,\, \bar{J}\,\vec{r}_0,\, \bar{J}^2\,\vec{r}_0,\, \cdots,\, \bar{J}^{m-1}\,\vec{r}_0\} \\ \vec{r}_0 &= \bar{J}\,\Delta\vec{W}_0^n + \vec{R}^n\,, \end{aligned} \tag{3.10}$$

such that the residual $||\bar{J}\,\Delta\vec{W}^n + \vec{R}^n||$ becomes a minimum. The parameter m specifies the dimension of the Krylov subspace, or in other words the number of *search directions* ($\bar{J}^i\,\vec{r}_0$). Since all directions have to be stored, m is usually chosen between 10 and 40, the higher number being necessary for poorly conditioned matrices (which arise in the simulation of turbulent flows, real gas, etc.). GMRES has to be restarted, if no convergence is achieved within m subiterations. The GMRES method requires significantly more memory than, e.g., Bi-CGSTAB or TFQMR, but it is more robust, smoothly converging and usually also faster. A very detailed comparison of the various methodologies can be found in [187].

Nevertheless, as with other conjugate gradient methods, preconditioning is absolutely essential for CFD problems. Here, we solve

$$(\bar{P}_L\bar{J})\,\Delta\vec{W}^n = -\bar{P}_L\,\vec{R}^n \quad ,\text{ or } \quad \bar{J}\,\bar{P}_R\,(\bar{P}_R^{-1}\Delta\vec{W}^n) = -\vec{R}^n \tag{3.11}$$

instead of the system in Eq. (3.9). The matrices $\bar{P}_L$ and $\bar{P}_R$ denote left and right preconditioners, respectively. The preconditioner should approximate $\bar{J}^{-1}$ as close as possible, in order to cluster the eigenvalues near unity. On the other hand, it should be of course easy to invert. One particularly efficient preconditioner is the *Incomplete Lower Upper* factorisation method [188] with zero fill-in (ILU(0)). For the discussion of different preconditioning techniques in connection with GMRES the reader is referred to [94], [189]-[192].

Since the GMRES method requires a considerable amount of computer memory for storing the search directions and possibly also the preconditioning matrix, it is a good idea to circumvent an explicit formation and storage of the flux Jacobian $\partial\vec{R}/\partial\vec{W}$. This is offered by the so-called *matrix-free* approach. The

idea is based on the observation that GMRES (and some other Krylov subspace methods) only employs matrix vector products of the form

$$\frac{\partial \vec{R}}{\partial \vec{W}} \Delta \vec{W}^n ,$$

which can be simply approximated by finite-differences as

$$\frac{\partial \vec{R}}{\partial \vec{W}} \Delta \vec{W}^n = \frac{\vec{R}(\vec{W} + \epsilon \Delta \vec{W}^n) - \vec{R}(\vec{W})}{\epsilon} , \tag{3.12}$$

thus requiring only residual evaluations. The parameter ϵ has to be chosen with some care, in order to minimise the numerical error (see, e.g., [193] or [194]). Another, and even more important, advantage of the matrix-free approach is that (numerically) accurate linearisation of a high-order residual $\vec{R}^n$ can easily be utilised in the implicit scheme. Hence, the quadratic convergence of Newton's scheme can be achieved at moderate costs. In this case we speak of *Newton-Krylov* approach [194]-[198], [107]. Practical experience indicates that from all Krylov subspace methods, GMRES is best suited for the matrix-free implementation [199]. An interesting possibility is to utilise the LU-SGS scheme as a preconditioner for the matrix-free GMRES method. Since the LU-SGS scheme does not also require an explicit storage of the flux Jacobian, the memory requirements can be reduced even further. The computational efficiency of this approach was recently demonstrated for 3-D inviscid and laminar flows on unstructured grids [200].

The convergence of an implicit scheme can also be enhanced by using multigrid. There are basically two possible ways. Firstly, we can employ multigrid inside an implicit scheme – as a solver for the linear equation system (3.9) arising at each time step, or as a preconditioner for one of the conjugate gradient methods [201], [202]. Secondly, the implicit scheme itself can serve as a smoother within the FAS multigrid method, which is applied directly to the governing equations [203]-[206], [175]. Some investigations show that at least for purely aerodynamic problems, rather "simple" implicit schemes (like Gauss-Seidel) combined with multigrid result in computationally more efficient solvers (in terms of CPU-time) than, e.g., GMRES [175], [195].

3.3 Turbulence Modelling

The solution of the governing equations (2.19) does not raise any fundamental difficulties in the case of inviscid or laminar flows. The simulation of turbulent flows, however, presents a significant problem. Despite the performance of modern supercomputers, a direct simulation of turbulence by the time-dependent Navier-Stokes equations (2.19), called *Direct Numerical Simulation* (DNS), is still possible only for rather simple flow cases at low Reynolds numbers (Re). The restrictions of the DNS become quite obvious when recalling that the number of grid points needed for sufficient spatial resolution scales as $Re^{9/4}$ and the CPU-time as Re^3. This does not mean that DNS is completely useless. It is an important tool for understanding the turbulent structures and the laminar-turbulent transition. DNS also plays a vital role in the development and calibration of new or improved turbulence models. However, in engineering applications, the effects of turbulence can be taken into account only approximately, using models of various complexities.

The first level of approximation is reached for the *Large-Eddy Simulation* (LES) approach. The development of LES is founded on the observation that the small scales of turbulent motion posses a more universal character than the large scales, which transport the turbulent energy. Thus, the idea is to resolve only the large eddies accurately and to approximate the effects of the small scales by relatively simple *subgrid-scale models*. Since LES requires significantly less grid points than DNS, the investigation of turbulent flows at much higher Reynolds numbers becomes feasible. But because LES is inherently three-dimensional and unsteady, it remains computationally still very demanding. Thus, LES is still far from becoming an engineering tool. However, LES is well suited for detailed studies of complex flow physics including massively separated unsteady flows, large scale mixing (e.g., fuel and oxidiser), aerodynamic noise, or for the investigation of flow control strategies. LES is also very promising for more accurate computations of flows in combustion chambers or engines, heat transfer and of rotating flows. An overview of research activities in LES was recently published in [207].

The next level of approximation is represented by the so-called *Reynolds-Averaged* Navier-Stokes equations (RANS). This approach, which was presented by Reynolds in 1895, is based on the decomposition of the flow variables into mean and fluctuating parts followed by time or ensemble averaging [208] (see also [209], [210]). In cases where the density is not constant, it is advisable to apply the *density (mass) weighted* or Favre decomposition [211], [212] to the velocity components. Otherwise, the averaged governing equations would become considerably more complicated due to additional correlations involving density fluctuations. It is common to assume that Morkovin's hypothesis [213] is valid, which states that the turbulence structure of boundary layers and wakes is not notably influenced by density fluctuations for Mach numbers below 5.

By inserting the decomposed variables into the Navier-Stokes equations (2.19) and averaging, we obtain formally the same equations for the mean variables with the exception of two additional terms. The tensor of the viscous

stresses is extended by one term – the *Reynolds-stress tensor* [208]

$$\overline{\overline{\tau}}_{ij}^{R} = -\overline{\rho}\,\widetilde{v_i'' v_j''}\,, \tag{3.13}$$

where v_i'', v_j'' denote the density-weighted fluctuating parts of the velocity components u, v, w; $^-$ and $^\sim$ stand for ensemble and density weighted averaging, respectively. The Reynolds-stress tensor represents the transport of mean momentum due to turbulent fluctuations. Furthermore, the diffusive heat flux $k\nabla T$ in the energy equation (cf. Eq. (2.8)) is enhanced by the so-called *turbulent heat-flux vector* [41]

$$\vec{F}_D^T = -\overline{\rho}\,\widetilde{h''\vec{v}\,''}\,. \tag{3.14}$$

Thus we can see that the solution of the Reynolds-averaged Navier-Stokes equations requires the modelling of the Reynolds stresses (3.13) and the turbulent heat flux (3.14). The advantages of this approach are that considerably coarser grids can be used compared to LES, and that stationary mean solution can be assumed (at least for attached or moderately separated flows). Clearly, both features significantly reduce the computational effort in comparison to LES or even DNS. Therefore, the RANS approach is very popular in engineering applications. Of course, because of the averaging procedure, no detailed information can be obtained about turbulent structures.

A large variety of turbulence models was devised to close the RANS equations and the research still continues. The models can be divided into *first-* and *second-order* closures, respectively.

The most complex, but also the most flexible, are second-order closure models. The *Reynolds-Stress Transport* (RST) model, which was first proposed by Rotta [214], solves modelled transport equations for the Reynolds-stress tensor. The partial differential equations for the six stress components have to be closed by one additional relation. Usually, an equation for the turbulent dissipation rate is employed. The RST models are able to account for strong nonlocal and history effects. Furthermore, they are able to capture the influence of streamline curvature or system rotation on the turbulent flow.

Closely related to the RST approach are the *Algebraic Reynolds-Stress* (ARS) models. They can be viewed as a combination of lower level models and the RST approach. The ARS models employ only two transport equations, mostly for the turbulent kinetic energy and the dissipation rate. The components of the Reynolds-stress tensor are related to the transport quantities by non-linear algebraic equations [215]. The ARS approach is capable of predicting rotational turbulent flows and secondary flows in channels with accuracy similar to the RST models. Detailed overviews of the RST and ARS models can be found in [216], [217].

Because of numerical problems with the RST and ARS models, which are primarily caused by the stiffness of the RST and the non-linearity of the ARS equations, first-order closures are more widely used in practice. In these models, the Reynolds stresses are expressed by means of a single scalar value, the so-called *turbulent eddy viscosity*. This approach is based on the *eddy viscosity*

hypothesis of Boussinesq [218], [219], which assumes a linear relationship between the turbulent shear stress and the mean strain rate, similar to laminar flow. Herewith, the dynamic viscosity μ in the viscous stress tensor (2.15) or in the governing equations (2.19) is replaced by the sum of a laminar and a turbulent component

$$\mu = \mu_L + \mu_T \,. \tag{3.15}$$

As described earlier, the laminar viscosity is calculated, for example, with the aid of the Sutherland formula (2.30). In analogy, the turbulent heat-flux vector (3.14) is modelled as

$$\vec{F}_D^T = -k_T \, \nabla T \,, \tag{3.16}$$

where k_T denotes the *turbulent thermal conductivity coefficient*. Hence, the thermal conductivity coefficient in Eq. (2.24) is evaluated as

$$k = k_L + k_T = c_p \left(\frac{\mu_L}{Pr_L} + \frac{\mu_T}{Pr_T} \right) . \tag{3.17}$$

The turbulent Prandtl number is in general assumed to be constant in the flow field ($Pr_T = 0.9$ for air). The coefficient of the turbulent eddy viscosity μ_T has to be determined with the aid of a turbulence model. The limitations of the eddy viscosity approach are given by the assumption of equilibrium between the turbulence and the mean strain field, and by the independence on system rotation. The accuracy of the eddy-viscosity based models can be significantly improved either by using correction terms [220], [221], or by employing *non-linear eddy viscosity* approaches [222]-[224].

The first-order closures can be categorised into *zero-*, *one-*, and *multiple-equation* models, corresponding to the number of transport equations they utilise. Within the zero-equation or, as they are also denoted, *algebraic models*, the turbulent eddy viscosity is calculated from empirical relations, which employ only local mean flow variables. Therefore, no history effects can be simulated, which prevents a reliable prediction of separated flows. The most popular algebraic model, which is still in use for some applications, was developed by Baldwin and Lomax [225].

History effects are taken into account by the one- and two-equation models, where the convection and the diffusion of turbulence is modelled by transport equations. The most widely used one-equation turbulence model is due to Spalart and Allmaras [226], which is based on an eddy-viscosity like variable. The model is numerically very stable and easy to implement on structured as well as unstructured grids.

In the case of the two-equation models, practically all approaches employ the transport equation for the turbulent kinetic energy. Among a large number of two-equation models, the $K\!-\!\varepsilon$ model of Launder and Spalding [227] and the $K-\omega$ model of Wilcox [228] are most often used in engineering applications. They offer a reasonable compromise between computational effort and accuracy. An interesting comparison between the Spalart-Allmaras model and various two-equation turbulence models was recently published in [229].

3.4 Initial and Boundary Conditions

Regardless of the numerical methodology chosen to solve the governing equations (2.19), we have to specify suitable *initial* and *boundary* conditions. The initial conditions determine the state of the fluid at the time $t = 0$, or at the first step of an iterative scheme. Clearly, the better (the closer to the solution) the initial guess will be, the faster the final solution will be obtained. Moreover, the probability of breakdown of the numerical solution process will be reduced correspondingly. Therefore, it is important that the initial solution satisfies at least the governing equations and the additional thermodynamic relations. A common practice in external aerodynamics consists of prescribing freestream values of pressure, density and velocity components (given as Mach number, angle of attack and sideslip angle) in the whole flow field. In turbomachinery, it is important to specify the flow directions in the complete domain to one's best knowledge. The same holds also for the pressure field. It is therefore quite worthwhile to employ lower-order approximations (like potential methods) to generate a physically meaningful initial guess.

Any numerical flow simulation considers only a certain part of the physical domain. The truncation of the computational domain creates artificial boundaries, where values of the physical quantities have to be specified. Examples are the farfield boundary in external aerodynamics; the inlet, outlet and the periodic boundary in the case of internal flows; and finally the symmetry plane. The main problem when constructing such boundary conditions is of course that the solution on the truncated domain should stay as close as possible to a solution which would be obtained for the whole physical domain. In the case of the farfield, inlet and outlet boundaries, *characteristic* boundary conditions [230]-[232] are often used in order to suppress the generation of non-physical disturbances in the flow field. But despite this, the farfield or the inlet and outlet boundaries may still not be placed too close to the object under consideration (wing, blade, etc.). Otherwise, the accuracy of the solution would be reduced. For external flows, when a lifting body is considered, it is possible to correct the flow variables at the farfield boundary using a single vortex centred at the airfoil or the wing [232]-[234]. In this way, the distance between the body and the farfield boundary can be significantly reduced without hampering the solution accuracy, or improving the accuracy for a given outer boundary position [157], [234], [235]. For internal flow problems, formulations for the inlet and outlet boundaries based on linearised Euler equations and Fourier series expansion of the perturbations were developed [236]-[238]. These formulations allow for a very close placement of the inlet and outlet boundaries to a blade without influencing the solution.

A different type of boundary condition is found when the surface of a body is exposed to the fluid. In the case of inviscid flow governed by the Euler equations (2.45), the appropriate boundary condition is to require the flow to be tangential to the surface, i.e.,

$$\vec{v} \cdot \vec{n} = 0 \qquad \text{at the surface.}$$

By contrast, for the Navier-Stokes equations no relative velocity between the surface and the fluid immediately at the surface is assumed – the so-called *noslip* boundary condition

$$u = v = w = 0 \qquad \text{at the surface.}$$

The treatment of walls becomes more involved in cases, where, e.g., a specified wall temperature distribution has to be met, or when the heat radiation has to be taken into account (see, e.g., [239], [240]).

Furthermore, boundary conditions have to be defined for surfaces where different fluids (e.g., air and water) meet together [241]-[244]. But apart from the physical boundary conditions and those imposed by truncating the flow domain, there can be boundaries generated by the numerical solution method itself. These are for example coordinate cuts and block or zonal boundaries [19]-[25].

The correct implementation of boundary condition is the crucial point of every flow solver. Not only the accuracy of the solution depends strongly on a proper physical and numerical treatment of boundaries, but also the robustness and the convergence speed are considerably influenced. More details of various important boundary conditions are presented in Chapter 8.

Bibliography

[1] Frymier, P.D.; Hassan, H.A.; Salas, M.D.: *Navier-Stokes Calculations Using Cartesian Grids: I Laminar Flows.* AIAA Journal, 26 (1988), pp. 1181-1188.

[2] De Zeeuw, D.; Powell, K.G.: *An Adaptively Refined Cartesian Mesh Solver for the Euler Equations.* J. Computational Physics, 104 (1993), pp. 56-68.

[3] Coirier, W.J.; Powell, K.G.: *An Accuracy Assessment of Cartesian-Mesh Approaches for the Euler Equations.* J. Computational Physics, 117 (1995), pp. 121-131.

[4] Richtmyer, R.D.; Morton, K.W.: *Difference Methods for Initial Value Problems.* Wiley-Interscience, 2nd edition, London, 1967.

[5] Volpe, G.: *Performance of Compressible Flow Codes at Low Mach number.* AIAA Journal, 31 (1993), pp. 49-56.

[6] Harlow, F.H.; Welch, J.E.: *Numerical Calculation of Time-Dependent Viscous Incompressible Flow with Free Surface.* Physics of Fluids, 8 (1965), pp. 2182-2189.

[7] Patankar, S.V.: *Numerical Heat Transfer and Fluid Flow.* McGraw-Hill, New York, 1980.

[8] Ho, Y.-H.; Lakshminarayana, B.: *Computation of Unsteady Viscous Flow Using a Pressure-Based Algorithm.* AIAA Journal, 31 (1993), pp. 2232-2240.

[9] Chorin, A.J.: *A Numerical Method for Solving Incompressible Viscous Flow Problems.* J. Computational Physics, 2 (1967), pp. 12-26.

[10] Turkel, E.: *Preconditioned Methods for Solving the Incompressible and Low Speed Compressible Equations.* J. Computational Physics, 72 (1987), pp. 277-298.

[11] Van Leer, B.; Lee, W.T.; Roe, P.L.: *Characteristic Time-Stepping or Local Preconditioning of the Euler Equations.* AIAA Paper 91-1552, 1991.

[12] Choi, Y.H.; Merkle, C.L.: *The Application of Preconditioning in Viscous Flows.* J. Computational Physics, 105 (1993), pp. 207-233.

[13] Turkel, E.: *Review of Preconditioning Methods for Fluid Dynamics.* Applied Numerical Mathematics, 12 (1993), pp. 257-284.

[14] Weiss, J.; Smith W.A.: *Preconditioning Applied to Variable and Constant Density Flows.* AIAA Journal, 33 (1995), pp. 2050-2057.

[15] Lee, D.: *Local Preconditioning of the Euler and Navier-Stokes Equations.* PhD Thesis, University of Michigan, 1996.

[16] Jespersen, D.; Pulliam, T.; Buning, P.: *Recent Enhancements to OVERFLOW.* AIAA Paper 97-0644, 1997.

[17] Merkle, C.L.; Sullivan, J.Y.; Buelow, P.E.O.; Venkateswaran, S.: *Computation of Flows with Arbitrary Equations of State.* AIAA Journal, 36 (1998), pp. 515-521.

[18] Sharov, D.; Nakahashi, K.: *Low Speed Preconditioning and LU-SGS Scheme for 3-D Viscous Flow Computations on Unstructured Grids.* AIAA Paper 98-0614, 1998.

[19] Lee, K.D.; Rubbert, P.E.: *Transonic Flow Computations Using Grid Systems with Block Structure.* Lecture Notes in Physics, 141 (1981), Springer Verlag, pp. 266-271.

[20] Rossow, C.-C.: *Efficient Computation of Inviscid Flow Fields Around Complex Configurations Using a Multiblock Multigrid Method.* Comm. in Appl. Num. Methods, 8 (1992), pp. 735-747.

[21] Kuerten, H.; Geurts, B.: *Compressible Turbulent Flow Simulation with a Multigrid Multiblock Method.* Proc. 6th Copper Mountain Conf. on Multigrid Methods, 1993.

[22] Rizzi, A.; et al.: *The Engineering of Multiblock/Multigrid Software for Navier Stokes Flows on Structured Meshes.* Computers & Fluids, 22 (1993), pp. 341-367.

[23] Enander, R.; Sterner, E.: *Analysis of Internal Boundary Conditions and Communication Strategies for Multigrid Multiblock Methods.* Dept. of Scientific Computing, Uppsala University, Sweden, Report No. 191, 1997.

[24] Rai, M.M.: *A Conservative Treatment of Zonal Boundaries for Euler Equations Calculations.* J. Computational Physics, 62 (1986), pp. 472-503.

[25] Kassies, A.; Tognaccini, R.: *Boundary Conditions for Euler Equations at Internal Block Faces of Multi-Block Domains Using Local Grid Refinement.* AIAA Paper 90-1590, 1990.

[26] Benek, J.A.; Buning, P.G.; Steger, J.L.: *A 3-D Chimera Grid Embedding Technique.* AIAA Paper 85-1523, 1985.

[27] Buning, P.G.; Chu, I.T.; Obayashi, S.; Rizk, Y.M.; Steger, J.L.: *Numerical Simulation of the Integrated Space Shuttle Vehicle in Ascent.* AIAA Paper 88-4359-CP, 1988.

[28] Chesshire, G.; Henshaw, W.D.: *Composite Overlapping Meshes for the Solution of Partial Differential Equations.* J. Computational Physics, 90 (1990), pp. 1-64.

[29] Pearce, D.G.; et al.: *Development of a Large Scale Chimera Grid System for the Space Shuttle Launch Vehicle.* AIAA Paper 93-0533, 1993.

[30] Kao, H.-J.; Liou, M.-S.; Chow, C.-Y.: *Grid Adaptation using Chimera Composite Overlapping Meshes.* AIAA Journal, 32 (1994), pp. 942-949.

[31] Thompson, J.F.; Weatherill, N.P.: *Aspects of Numerical Grid Generation: Current Science and Art.* AIAA Paper 93-3539, 1993.

[32] Nakahashi, K.: *FDM-FEM Zonal Approach for Computations of Compressible Viscous Flows.* Lecture Notes in Physics, 264 (1986), Springer Verlag, pp. 494-498.

[33] Nakahashi, K.: *A Finite-Element Method on Prismatic Elements for the Three-Dimensional Navier-Stokes Equations.* Lecture Notes in Physics, 323 (1989), Springer Verlag, pp. 434-438.

[34] Holmes, D.G.; Connell, S.D.: *Solution of the Two-Dimensional Navier-Stokes Equations on Unstructured Adaptive Grids.* AIAA Paper 89-1932, 1989.

[35] Peace, A.J.; Shaw, J.: *The Modelling of Aerodynamic Flows by Solution of the Euler Equations on Mixed Polyhedral Grids.* Int. J. Num. Meth. Eng., 35 (1992), pp. 2003-2029.

[36] Kallinderis, Y.; Khawaja, A.; McMorris, H.: *Hybrid Prismatic/Tetrahedral Grid Generation for Viscous Flows Around Complex Geometries.* AIAA Journal, 34 (1996), pp. 291-298.

[37] McMorris, H.; Kallinderis, Y.: *Octree-Advancing Front Method for Generation of Unstructured Surface and Volume Meshes.* AIAA Journal, 35 (1997), pp. 976-984.

[38] Peraire, J.; Morgan, K.: *Unstructured Mesh Generation for 3-D Viscous Flow.* AIAA Paper 98-3010, 1998.

[39] Marcum, D.L.; Gaither, J.A.: *Mixed Element Type Unstructured Grid Generation for Viscous Flow Applications.* AIAA Paper 99-3252, 1999.

[40] Mavriplis, D.J.: *Unstructured Grid Techniques.* Annu. Rev. Fluid. Mech., 29 (1997), pp. 473-514.

[41] Hirsch, C.: *Numerical Computation of Internal and External Flows.* Vols. 1 and 2, John Wiley and Sons, 1988.

[42] McDonald, P.W.: *The Computation of Transonic Flow through Two-Dimensional Gas Turbine Cascades.* ASME Paper 71-GT-89, 1971.

[43] Turner, M.J.; Clough, R.W.; Martin, H.C.; Topp, L.P.: *Stiffness and deflection analysis of complex structures.* J. Aeronautical Society, 23 (1956), p. 805.

[44] Zienkiewicz, O.C.; Taylor, R.L.: *The Finite Element Method.* 4th edition, McGraw-Hill, Maidenhead, 1991.

[45] Pironneau, O.: *Finite Element Methods for Fluids.* John Wiley, Chichester, 1989.

[46] Hassan, O.; Probert, E.J.; Morgan, K.; Peraire, J.: *Adaptive finite element methods for transient compressible flow problems.* In Brebbia, C.A. and Aliabadi, M.H., editors, *Adaptive Finite and Boundary Element Methods,* Elsevier Applied Science, London, 1993, pp. 119-160.

[47] Reddy, J.N.; Gartling, D.K.: *The Finite Element Method in Heat Transfer and Fluid Dynamics.* C.R.C Press, 1994.

[48] Patera, A.T.: *A Spectral Element Method for Fluid Dynamics; Laminar Flow in a Channel Expansion.* J. Computational Physics, 54 (1984), pp. 468-488.

[49] Maday, Y.; Patera, A.T.: *Spectral Element Methods for the Incompressible Navier-Stokes Equations.* ASME, State of the art surveys on Computational Mechanics, 1987, pp. 71-143.

[50] Henderson, R.D.; Karniadakis, G.: *Unstructured Spectral Element Methods for Simulation of Turbulent Flows.* J. Computational Physics, 122 (1995), pp. 191-217.

[51] Henderson, R.D.; Meiron, D.I.: *Dynamic Refinement Algorithms for Spectral Element Methods.* AIAA Paper 97-1855, 1997.

[52] Canuto, C.; Hussaini, M.Y.; Quarteroni, A.; Zang, T.A.: *Spectral Methods in Fluid Dynamics.* Springer Verlag, Berlin, 1987.

[53] Batina, J.T.: *A Gridless Euler/Navier-Stokes Solution Algorithm for Complex-Aircraft Applications.* AIAA Paper 93-0333, 1993.

[54] Shih, S.-C.; Lin, S.-Y.: *A Weighting Least Squares Method for Euler and Navier-Stokes Equations.* AIAA Paper 94-0522, 1994.

[55] Barth, T.J.: *Numerical Aspects of Computing High-Reynolds Number Flows on Unstructured Meshes.* AIAA Paper 91-0721, 1991.

[56] Mavriplis, D.J.; Venkatakrishnan, V.: *A Unified Multigrid Solver for the Navier-Stokes Equations on Mixed Element Meshes.* ICASE Report No. 95-53, 1995.

[57] Braaten, M.E.; Connell, S.D.: *Three Dimensional Unstructured Adaptive Multigrid Scheme for the Navier-Stokes Equations.* AIAA Journal, 34 (1996), pp. 281-290.

[58] Haselbacher, A.C.; McGuirk, J.J.; Page, G.J.: *Finite Volume Discretisation Aspects for Viscous Flows on Mixed Unstructured Grids.* AIAA Paper 97-1946, 1997; also AIAA Journal, 37 (1999), pp. 177-184.

[59] Crumpton, P.I.; Moiner, P.; Giles, M.B.: *An Unstructured Algorithm for High Reynolds Number Flows on Highly-Stretched Grids.* 10th Int. Conf. Num. Meth. for Laminar and Turbulent Flows, Swansea, England, July 21-25, 1997.

[60] Weiss, J.M.; Maruszewski, J.P.; Smith, W.A.: *Implicit Solution of Preconditioned Navier-Stokes Equations Using Algebraic Multigrid.* AIAA Journal, 37 (1999), pp. 29-36.

[61] Jameson, A.; Schmidt, W.; Turkel, E.: *Numerical Solutions of the Euler Equations by Finite Volume Methods Using Runge-Kutta Time-Stepping Schemes.* AIAA Paper 81-1259, 1981.

[62] Jameson, A.; Baker, T.J.; Weatherill, N.P.: *Calculation of Inviscid Transonic Flow over a Complete Aircraft.* AIAA Paper 86-0103, 1986.

[63] Swanson, R.C.; Turkel, E.: *On Central Difference and Upwind Schemes.* J. Computational Physics, 101 (1992), pp. 292-306.

[64] Haselbacher, A.; Blazek, J.: *On the Accurate and Efficient Discretisation of the Navier-Stokes Equations on Mixed Grids.* AIAA Paper 99-3363, 1999; also AIAA Journal, 38 (2000), pp. 2094-2102.

[65] Steger J.L.; Warming, R.F.: *Flux Vector Splitting of the Inviscid Gasdynamic Equations with Application to Finite Difference Methods.* J. Computational Physics, 40 (1981), pp. 263-293.

[66] Van Leer, B.: *Flux Vector Splitting for the Euler Equations.* Proc. 8th Int. Conf. on Numerical Methods in Fluid Dynamics, Springer Verlag, 1982, pp. 507-512; also ICASE Report 82-30, 1982.

[67] Liou, M.-S.; Steffen, C.J. Jr.: *A New Flux Splitting Scheme.* NASA TM-104404, 1991; also J. Computational Physics, 107 (1993), pp. 23-39.

[68] Liou, M.-S.: *A Sequel to AUSM: AUSM+.* J. Computational Physics, 129 (1996), pp. 364-382.

[69] Jameson, A.: *Positive Schemes and Shock Modelling for Compressible Flow.* Int. J. Numerical Methods in Fluids, 20 (1995), pp. 743-776.

[70] Tatsumi, S.; Martinelli, L.; Jameson, A.: *A New High Resolution Scheme for Compressible Viscous Flow with Shocks.* AIAA Paper 95-0466, 1995.

[71] Edwards, J.R.: *A Low-Diffusion Flux-Splitting Scheme for Navier-Stokes Calculations.* Computers & Fluids, 26 (1997), pp. 653-659.

[72] Rossow, C.-C.: *A Simple Flux Splitting Scheme for Compressible Flows.* Proc. 11th DGLR-Fach-Symposium, Berlin, Germany, November 10-12, 1998.

[73] Rossow, C.-C.: *A Flux Splitting Scheme for Compressible and Incompressible Flows.* AIAA Paper 99-3346, 1999.

[74] Godunov, S.K.: *A Difference Scheme for Numerical Computation Discontinuous Solution of Hydrodynamic Equations.* Math. Sbornik (in Russian), 47 (1959), pp. 271-306; translated US Joint Publ. Res. Service, JPRS 7226, 1969.

[75] Osher, S.; Solomon, F.: *Upwind Difference Schemes for Hyperbolic Systems of Conservation Laws.* Math. Comp., 38 (1982), pp. 339-374.

[76] Roe, P.L.: *Approximate Riemann Solvers, Parameter Vectors, and Difference Schemes.* J. Computational Physics, 43 (1981), pp. 357-372.

[77] Barth, T.J.; Jespersen, D.C.: *The Design and Application of Upwind Schemes on Unstructured Meshes.* AIAA Paper 89-0366, 1989.

[78] Harten, A.: *High Resolution Schemes for Hyperbolic Conservation Laws.* J. Computational Physics, 49 (1983), pp. 357-393.

[79] Yee, H.C.; Harten, A.: *Implicit TVD Schemes for Hyperbolic Conservation Laws in Curvilinear Coordinates.* AIAA Journal, 25 (1987), pp. 266-274.

[80] Yee, H.C.: *Construction of Implicit and Explicit Symmetric TVD Schemes and Their Applications.* J. Computational Physics, 68 (1987), pp. 151-179.

[81] Harten, A.; Engquist, B.; Osher, S.; Chakravarthy, S.: *Uniformly High Order Accurate Essentially Non-Oscillatory Schemes III.* J. Computational Physics, 71 (1987), pp. 231-303; also ICASE Report No. 86-22, 1986.

[82] Casper, J.; Atkins, H.L.: *A Finite-Volume High-Order ENO Scheme for Two-Dimensional Hyperbolic Systems.* J. Computational Physics, 106 (1993), pp. 62-76.

[83] Godfrey, A.G.; Mitchell, C.R.; Walters, R.W.: *Practical Aspects of Spatially High-Order Accurate Methods.* AIAA Journal, 31 (1993), pp. 1634-1642.

[84] Abgrall, R.; Lafon, F.C.: *ENO Schemes on Unstructured Meshes.* VKI Lecture Series 1993-04, 1993.

[85] Ollivier-Gooch, C.F.: *High-Order ENO Schemes for Unstructured Meshes Based on Least-Squares Reconstruction.* AIAA Paper 97-0540, 1997.

[86] Stanescu, D.; Habashi, W.G.: *Essentially Nonoscillatory Euler Solutions on Unstructured Meshes Using Extrapolation.* AIAA Journal, 36 (1998), pp. 1413-1416.

[87] Roe, P.L.: *Discrete Models for the Numerical Analysis of Time-Dependent Multidimensional Gas Dynamics.* J. Computational Physics, 63 (1986), pp. 458-476.

[88] Powell, K.G.; van Leer, B.; Roe, P.L.: *Towards a Genuinely Multi-Dimensional Upwind Scheme.* VKI Lecture Series 1990-03, 1990.

[89] Sidilkover, D.: *A Genuinely Multidimensional Upwind Scheme and Efficient Multigrid Solver for the Compressible Euler Equations.* ICASE Report, No. 94-84, 1994.

[90] Struijs, R.; Roe, P.L.; Deconinck, H.: *Fluctuation Splitting Schemes for the 2D Euler Equations.* VKI Lecture Series 1991-01, 1991.

[91] Paillére, H.; Deconinck, H.; Roe, P.L.: *Conservative Upwind Residual-Distribution Schemes Based on the Steady Characteristics of the Euler Equations.* AIAA Paper 95-1700, 1995.

[92] Issman, E.; Degrez, G.; Deconinck, H.: *Implicit Upwind Residual-Distribution Euler and Navier-Stokes Solver on Unstructured Meshes.* AIAA Journal, 34 (1996), pp. 2021-2028.

[93] Van der Weide, E.; Deconinck, H.: *Compact Residual-Distribution Scheme Applied to Viscous Flow Simulations.* VKI Lecture Series 1998-03, 1998.

[94] Venkatakrishnan, V.: *Preconditioned Conjugate Gradient Methods for the Compressible Navier-Stokes Equations.* AIAA Journal, 29 (1991), pp. 1092-1110.

[95] Venkatakrishnan, V.: *On the Accuracy of Limiters and Convergence to Steady State Solutions.* AIAA Paper 93-0880, 1993.

[96] Venkatakrishnan, V.: *Convergence to Steady-State Solutions of the Euler Equations on Unstructured Grids with Limiters.* J. Computational Physics, 118 (1995), pp. 120-130.

[97] Vinokur, M.: *Flux Jacobian Matrices and Generalized Roe Average for an Equilibrium Real Gas.* NASA CR-177512, 1988.

[98] Vinokur, M.; Liu, Y.: *Equilibrium Gas Flow Computations: II. An Analysis of Numerical Formulations of Conservation Laws.* AIAA Paper 88-0127, 1988.

[99] Vinokur, M.; Montagné, J.-L.: *Generalized Flux-Vector Splitting and Roe Average for an Equilibrium Real Gas.* J. Computational Physics, 89 (1990), pp. 276-300.

[100] Vinokur, M.; Liu, Y.: *Nonequilibrium Flow Computations: I. An Analysis of Numerical Formulations of Conservation Laws.* NASA CR-177489, 1988.

[101] Molvik, G.A.; Merkle, C.L.: *A Set of Strongly Coupled, Upwind Algorithms for Computing Flows in Chemical Nonequilibrium.* AIAA Paper 89-0199, 1989.

[102] Liu, Y.; Vinokur, M.: *Upwind Algorithms for General Thermo-Chemical Nonequilibrium Flows.* AIAA Paper 89-0201, 1989.

[103] Grossman, B.; Cinnella, P.: *Flux-Split Algorithms for Flows with Non-equilibrium Chemistry and Vibrational Relaxation.* J. Computational Physics, 88 (1990), pp. 131-168.

[104] Shuen, J.-S.; Liou, M.-S.; Van Leer, B.: *Inviscid Flux-Splitting Algorithms for Real Gases with Non-equilibrium Chemistry.* J. Computational Physics, 90 (1990), pp. 371-395.

[105] Slomski, J.F.; Anderson, J.D.; Gorski, J.J.: *Effectivness of Multi-grid in Accelerating Convergence of Multidimensional Flows in Chemical Nonequilibrium.* AIAA Paper 90-1575, 1990.

[106] Yu, S.-T.; Chang, S.-C.; Jorgenson, P.C.E.; Park, S.-J.; Lai, M.-C.: *Basic Equations of Chemically Reactive Flow for Computational Fluid Dynamics.* AIAA Paper 98-1051, 1998.

[107] Venkatakrishnan, V.: *Implicit Schemes and Parallel Computing in Unstructured Grid CFD.* ICASE Report No. 95-28, 1995.

[108] Venkatakrishnan, V.; Mavriplis, D.J.: *Implicit Method for the Computation of Unsteady Flows on Unstructured Grids.* J. Computational Physics, 127 (1996), pp. 380-397.

[109] Van Leer, B.; Tai, C.H.; Powell, K.G.: *Design of Optimally Smoothing Multi-Stage Schemes for the Euler Equations.* AIAA Paper 89-1933, 1989.

[110] Chang, H.T.; Jiann, H.S.; van Leer, B.: *Optimal Multistage Schemes for Euler Equations with Residual Smoothing.* AIAA Journal, 33 (1995), pp. 1008-1016.

[111] Shu, C.W.; Osher, S.: *Efficient Implementation of Essentially Non-Oscillatory Shock Capturing Schemes.* J. Computational Physics, 77 (1988), pp. 439-471.

[112] Lafon, A.; Yee, H.C.: *On the Numerical Treatment of Nonlinear Source Terms in Reaction-Convection Equations.* AIAA Paper 92-0419, 1992.

[113] Van Leer, B.; Lee, W.T.; Roe, P.: *Characteristic Time-Stepping or Local Preconditioning of the Euler Equations.* AIAA Paper 91-1552, 1991.

[114] Riemslagh, K.; Dick, E.: *A Multigrid Method for Steady Euler Equations on Unstructured Adaptive Grids.* Proc. 6th Copper Mountain Conf. on Multigrid Methods, NASA Conf. Publ. 3224 (1993), pp. 527-542.

[115] Morano, E.; Dervieux, A.: *Looking for O(N) Navier-Stokes Solutions on Non-Structured Meshes.* Proc. 6th Copper Mountain Conf. on Multigrid Methods, NASA Conf. Publ. 3224 (1993), pp. 449-464.

[116] Ollivier-Gooch, C.F.: *Towards Problem-Independent Multigrid Convergence Rates for Unstructured Mesh Methods.* Proc. 6th Int. Symp. CFD, Lake Tahoe, NV, 1995.

[117] Jameson, A.: *Solution of the Euler Equations by a Multigrid Method.* Applied Mathematics and Computation, 13 (1983), pp. 327-356.

[118] Jameson, A.: *Computational Transonics.* Comm. on Pure and Appl. Math., Vol. XLI (1988), pp. 507-549.

[119] Blazek, J.; Kroll, N.; Radespiel, R.; Rossow, C.-C.: *Upwind Implicit Residual Smoothing Method for Multi-Stage Schemes.* AIAA Paper 91-1533, 1991.

[120] Blazek, J.; Kroll, N.; Rossow, C.-C.: *A Comparison of Several Implicit Smoothing Methods.* Proc. ICFD Conf. on Numerical Meth. for Fluid Dynamics, Reading, 1992, pp. 451-460.

[121] Enander, E.: *Improved Implicit Residual Smoothing for Steady State Computations of First-Order Hyperbolic Systems.* J. Computational Physics, 107 (1993), pp. 291-296.

[122] Enander, E.; Sjögreen, B.: *Implicit Explicit Residual Smoothing for Upwind Schemes.* Internal Report 96-179, Department of Scientific Computing, Uppsala University, Sweden, 1996.

[123] Fedorenko, R.P.: *A Relaxation Method for Solving Elliptic Difference Equations.* U.S.S.R. Computational Math. and Math. Phys., Vol. 1, No. 5 (1962), pp. 1092-1096.

[124] Bakhvalov, N.S.: *On the Convergence of a Relaxation Method with Natural Constraints on the Elliptic Operator.* U.S.S.R. Computational Math. and Math. Phys., Vol. 6, No. 5 (1966), pp. 101-135.

[125] Brandt, A.: *Multi-Level Adaptive Solutions to Boundary-Value Problems.* Math. Comp., 31 (1977), pp. 333-390.

[126] Brandt, A.: *Guide to Multigrid Development.* Multigrid Methods I, Lecture Notes in Mathematics, No. 960, Springer Verlag, 1981.

[127] Jameson, A.: *Multigrid Algorithms for Compressible Flow Calculations.* Multigrid Methods II, Lecture Notes in Mathematics, No. 1228, Springer Verlag, 1985, pp. 166-201.

[128] Martinelli, L.: *Calculation of Viscous Flows with a Multigrid Method.* Ph.D. Thesis, Dept. of Mech. and Aerospace Eng., Princeton University, 1987.

[129] Mulder, W.A.: *A New Multigrid Approach to Convection Problems.* J. Computational Physics, 83 (1989), pp. 303-323.

[130] Koren, B.: *Multigrid and Defect Correction for the Steady Navier-Stokes Equations.* J. Computational Physics, 87 (1990), pp. 25-46.

[131] Turkel, E.; Swanson, R.C.; Vatsa, V.N.; White, J.A.: *Multigrid for Hypersonic Viscous Two- and Three-Dimensional Flows.* AIAA Paper 91-1572, 1991.

[132] Radespiel, R.; Swanson, R.C.: *Progress with Multigrid Schemes for Hypersonic Flow Problems.* ICASE Report, No. 91-89, 1991; also J. Computational Physics, 116 (1995), pp. 103-122.

[133] Arnone, A.; Pacciani, R.: *Rotor-Stator Interaction Analysis Using the Navier-Stokes Equations and a Multigrid Method.* Transactions ASME: Journal of Turbomachinery, 118 (1996), pp. 679-689.

[134] Mavriplis, D.J.: *Three-Dimensional Multigrid for the Euler Equations.* AIAA Journal, 30 (1992), pp. 1753-1761.

[135] Peraire, J.; Peiró, J.; Morgan, K.: *A 3D Finite-Element Multigrid Solver for the Euler Equations.* AIAA Paper 92-0449, 1992.

[136] Lallemand, M.; Steve, H.; Dervieux, A.: *Unstructured Multigridding by Volume Agglomeration: Current Status.* Computers & Fluids, 21 (1992), pp. 397-433.

[137] Crumpton, P.I.; Giles, M.B.: *Aircraft Computations Using Multigrid and an Unstructured Parallel Library.* AIAA Paper 95-0210, 1995.

[138] Ollivier-Gooch, C.F.: *Multigrid Acceleration of an Upwind Euler Solver on Unstructured Meshes.* AIAA Journal, 33 (1995), pp. 1822-1827.

[139] Mavriplis, D.J.: *Multigrid Techniques for Unstructured Meshes.* ICASE Report No. 95-27, 1995.

[140] Mavriplis, D.J.; Venkatakrishnan, V.: *A 3D Agglomeration Multigrid Solver for the Reynolds-Averaged Navier-Stokes Equations on Unstructured Meshes.* Int. Journal for Numerical Methods in Fluids 23 (1996), pp. 527-544.

[141] Mavriplis, D.J.: *Directional Agglomeration Multigrid Techniques for High Reynolds Number Viscous Flow Solvers.* AIAA Paper 98-0612, 1998.

[142] Mavriplis, D.J.: *Multigrid Strategies for Viscous Flow Solvers on Anisotropic Unstructured Meshes.* J. Computational Physics, 145 (1998), pp. 141-165.

[143] Okamoto, N.; Nakahashi, K.; Obayashi, S.: *A Coarse Grid Generation Algorithm for Agglomeration Multigrid Method on Unstructured Grids.* AIAA Paper 98-0615, 1998.

[144] Roberts, T.W.; Sidilkover, D.; Swanson, R.C.: *Textbook Multigrid Efficiency for the Steady Euler Equations.* AIAA Paper 97-1949, 1997.

[145] Pulliam, T.H.: *Time Accuracy and the Use of Implicit Methods.* AIAA Paper 93-3360, 1993.

[146] Arnone, A.; Liou, M.-S.; Povinelli, L.A.: *Multigrid Time-Accurate Integration of Navier-Stokes Equations.* AIAA Paper 93-3361, 1993.

[147] Alonso, J.; Martinelli, L.; Jameson, A.: *Multigrid Unsteady Navier-Stokes Calculations with Aeroelastic Applications.* AIAA Paper 95-0048, 1995.

[148] George, A.; Liu, J.W.: *Computer Solution of Large Sparse Positive Definite Systems.* Prentice Hall Series in Comput. Math., Englewood Cliffs, N.J., 1981.

[149] Pothen, A.; Simon, H.D.; Liou, K.P.: *Partitioning Sparse Matrices with Eigenvectors of Graphs.* SIAM J. Matrix Anal. Appl., 11 (1990), pp. 430-452.

[150] Bender, E.E.; Khosla, P.K.: *Solution of the Two-Dimensional Navier-Stokes Equations Using Sparse Matrix Solvers.* AIAA Paper 87-0603, 1987.

[151] Venkatakrishnan, V.: *Newton Solution of Inviscid and Viscous Problems.* AIAA Journal, 27 (1989), pp. 885-891.

[152] Beam, R.M.; Bailey, H.S.: *Viscous Computations Using a Direct Solver.* Computers & Fluids, 18 (1990), pp. 191-204.

[153] Vanden, K.J.; Whitfield, D.L.: *Direct and Iterative Algorithms for the Three-Dimensional Euler Equations.* AIAA Journal, 33 (1995), pp. 851-858.

[154] Venkatakrishnan, V.; Barth, T.J.: *Application of Direct Solvers to Unstructured Meshes for the Euler and Navier-Stokes Equations Using Upwind Schemes.* AIAA Paper 89-0364, 1989.

[155] Briley, W.R.; McDonald, H.: *Solution of the Multi-Dimensional Compressible Navier-Stokes Equations by a Generalized Implicit Method.* J. Computational Physics, 24 (1977), pp. 372-397.

[156] Beam, R.; Warming, R.F.: *An Implicit Factored Scheme for the Compressible Navier-Stokes Equations.* AIAA Journal, 16 (1978), pp. 393-402.

[157] Pulliam, T.H.; Steger, J.L.: *Recent Improvements in Efficiency, Accuracy and Convergence for Implicit Approximate Factorization Scheme.* AIAA Paper 85-0360, 1985.

[158] Rosenfeld, M.; Yassour, Y.: *The Alternating Direction Multi-Zone Implicit Method.* J. Computational Physics, 110 (1994), pp. 212-220.

[159] Golub, G.H.; Van Loan, C.F.: *Matrix Computations.* The Johns Hopkins University Press, Baltimore, Maryland, 1983.

[160] Chakravarthy, S.R.: *Relaxation Methods for Unfactored Implicit Schemes.* AIAA Paper 84-0165, 1984.

[161] Napolitano, M.; Walters, R.W.: *An Incremental Line Gauss-Seidel Method for the Incompressible and Compressible Navier-Stokes Equations.* AIAA Paper 85-0033, 1985.

[162] Thomas, J.L.; Walters, R.W.: *Upwind Relaxation Algorithms for the Navier-Stokes Equations.* AIAA Journal, 25 (1987), pp. 527-534.

[163] Jenssen, C.B.: *Implicit Multiblock Euler and Navier-Stokes Calculations.* AIAA Journal, 32 (1994), pp. 1808-1814.

[164] Yoon, S.; Jameson, A.: *Lower-Upper Implicit Schemes with Multiple Grids for the Euler Equations.* AIAA Paper 86-0105; also AIAA Journal, 7 (1987), pp. 929-935.

[165] Yoon, S.; Jameson, A.: *An LU-SSOR Scheme for the Euler and Navier-Stokes Equations.* AIAA Paper 87-0600; also AIAA Journal, 26 (1988), pp. 1025-1026.

[166] Rieger, H.; Jameson, A.: *Solution of Steady Three-Dimensional Compressible Euler and Navier-Stokes Equations by an Implicit LU Scheme.* AIAA Paper 88-0619, 1988.

[167] Shuen, J.-S.: *Upwind Differencing and LU Factorization for Chemical Non-Equilibrium Navier-Stokes Equations.* J. Computational Physics, 99 (1992), pp. 233-250.

[168] Blazek, J.: *Investigations of the Implicit LU-SSOR Scheme.* DLR Research Report, No. 93-51, 1993.

[169] Fezoui, L.; Stoufflet, B.: *A Class of Implicit Upwind Schemes for Euler Simulations with Unstructured Meshes.* J. Computational Physics, 84 (1989), pp. 174-206.

[170] Karman, S.L.: *Development of a 3D Unstructured CFD Method.* PhD thesis, The University of Texas at Arlington, 1991.

[171] Batina, J.T.: *Implicit Upwind Solution Algorithms for Three-Dimensional Unstructured Meshes.* AIAA Journal, 31 (1993), pp. 801-805.

[172] Slack, D.C.; Whitaker, D.L.; Walters, R.W.: *Time Integration Algorithms for the Two-Dimensional Euler Equations on Unstructured Meshes.* AIAA Journal, 32 (1994), pp. 1158-1166.

[173] Anderson, W.K.: *A Grid Generation and Flow Solution Method for the Euler Equations on Unstructured Grids.* J. Computational Physics, 110 (1994), pp. 23-38.

[174] Anderson, W.K.; Bonhaus, D.L.: *An Implicit Upwind Algorithm for Computing Turbulent Flows on Unstructured Grids.* Computers & Fluids, 23 (1994), pp. 1-21.

[175] Anderson, W.K.; Rausch, R.D.; Bonhaus, D.L.: *Implicit / Multigrid Algorithms for Incompressible Turbulent Flows on Unstructured Grids.* J. Computational Physics, 128 (1996), pp. 391-408; also AIAA Paper 95-1740, 1995.

[176] Tomaro, R.F.; Strang, W.Z.; Sankar, L.N.: *An Implicit Algorithm for Solving Time Dependent Flows on Unstructured Grids.* AIAA Paper 97-0333, 1997.

[177] Kano, S.; Nakahashi, K.: *Navier-Stokes Computations of HSCT Off-Design Aerodynamics Using Unstructured Hybrid Grids.* AIAA Paper 98-0232, 1998.

[178] Hassan, O.; Morgan, K.; Peraire, J.: *An Adaptive Implicit / Explicit Finite Element Scheme for Compressible High Speed Flows.* AIAA Paper 89-0363, 1989.

[179] Hassan, O.; Morgan, K.; Peraire, J.: *An Implicit Finite Element Method for High Speed Flows.* AIAA Paper 90-0402, 1990.

[180] Gibbons, A.: *Algorithmic Graph Theory.* Cambridge University Press, New York, NY, 1985.

[181] Löhner, R.; Martin, D.: *An Implicit Linelet-Based Solver for Incompressible Flows.* AIAA Paper 92-0668, 1992.

[182] Hestenes, M.R.; Stiefel, E.L.: *Methods of Conjugate Gradients for Solving Linear Systems.* J. Res. Nat. Bur. Stand., 49 (1952), p. 409.

[183] Sonneveld, P.: *CGS, A Fast Lanczos-Type Solver for Nonsymmetric Linear Systems.* SIAM J. Scientific Statistics and Computing, 10 (1989), pp. 36-52.

[184] Van der Vorst, H.A.: *BiCGSTAB: A Fast and Smoothly Converging Variant of Bi-CG for the Solution of Nonsymmetric Linear Systems.* SIAM J. Scientific and Statistical Computing, 13 (1992), pp. 631-644.

[185] Freund, R.W.: *A Transpose-Free Quasi-Minimal Residual Algorithm for Non-Hermitian Linear Systems.* SIAM J. Scientific Computing, 14 (1993), pp. 470-482.

[186] Saad, Y.; Schulz, M.H.: *GMRES: A Generalized Minimum Residual Algorithm for Solving Nonsymmetric Linear Systems.* SIAM J. Scientific and Statistical Computing, 7 (1986), pp. 856-869.

[187] Meister, A.: *Comparison of Different Krylov Subspace Methods Embedded in an Implicit Finite Volume Scheme for the Computation of Viscous and Inviscid Flow Fields on Unstructured Grids.* J. Computational Physics, 140 (1998), pp. 311-345.

[188] Meijerink, J.A.; Van der Vorst, H.A.: *Guidelines for Usage of Incomplete Decompositions in Solving Sets of Linear Equations as they occur in Practical Problems.* J. Computational Physics, 44 (1981), pp. 134-155.

[189] Wigton, L.B.; Yu, N.J.; Young D.P.: *GMRES Acceleration of Computational Fluid Dynamics Codes.* AIAA Paper 85-1494, 1985.

[190] Kadioğlu, M.; Mudrick, S.: *On the Implementation of the GMRES(m) Method to Elliptic Equations in Meteorology.* J. Computational Physics, 102 (1992), pp. 348-359.

[191] Venkatakrishnan, V.; Mavriplis, D.J.: *Implicit Solvers for Unstructured Meshes.* J. Computational Physics, 105 (1993), pp. 83-91.

[192] Ajmani, K.; Ng, W.-F.; Liou, M.-S.: *Preconditioned Conjugate Gradient Methods for Low Speed Flow Calculations.* AIAA Paper, 93-0881, 1993.

[193] Dennis, J.E.; Schnabel, R.B.: *Numerical Methods for Unconstrained Optimization and Nonlinear Equations.* Prentice-Hall, Englewood Cliffs, NJ, 1983.

[194] Brown, P.N.; Saad, Y.: *Hybrid Krylov Methods for Nonlinear Systems of Equations.* SIAM J. Scientific and Statistical Computing, 11 (1990), pp. 450-481.

[195] Nielsen, E.J.; Anderson, W.K.; Walters, R.W.; Keyes, D.E.: *Application of Newton-Krylov Methodology to a Three-Dimensional Unstructured Euler Code.* AIAA Paper 95-1733, 1995.

[196] Cai, X.; Keyes, D.E.; Venkatakrishnan, V.: *Newton-Krylov-Schwarz: An Implicit Solver for CFD.* ICASE Report 95-87, 1995.

[197] Tidriri, M.D.: *Schwarz-Based Algorithms for Compressible Flows.* ICASE Report 96-4, 1996.

[198] Zingg, D.; Pueyo, A.: *An Efficient Newton-GMRES Solver for Aerodynamic Computations.* AIAA Paper 97-1955, 1997.

[199] McHugh, P.R.; Knoll, D.A.: *Inexact Newton's Method Solutions to the Incompressible Navier-Stokes and Energy Equations Using Standard and Matrix-Free Implementations.* AIAA Paper 93-3332, 1993.

[200] Luo, H.; Baum, J.D.; Löhner, R.: *A Fast, Matrix-Free Implicit Method for Compressible Flows on Unstructured Grids.* AIAA Paper 99-0936, 1999.

[201] Raw, M.: *Robustness of Coupled Algebraic Multigrid for the Navier-Stokes Equations.* AIAA Paper 96-0297, 1996.

[202] Brieger, L.; Lecca, G.: *Parallel Multigrid Preconditioning of the Conjugate Gradient Method for Systems of Subsurface Hydrology.* J. Computational Physics, 142 (1998), pp. 148-162.

[203] Yoon, S.; Kwak, D.: *Multigrid Convergence of an Implicit Symmetric Relaxation Scheme.* AIAA Paper 93-3357, 1993.

[204] Blazek, J.: *A Multigrid LU-SSOR Scheme for the Solution of Hypersonic Flow Problems.* AIAA Paper 94-0062, 1994.

[205] Oosterlee, C.W.: *A GMRES-Based Plane Smoother in Multigrid to Solve 3D Anisotropic Fluid Flow Problems.* J. Computational Physics, 130 (1997), pp. 41-53.

[206] Gerlinger, P.; Stoll, P.; Brüggemann, D.: *An Implicit Multigrid Method for the Simulation of Chemically Reacting Flows.* J. Computational Physics, 146 (1998), pp. 322-345.

[207] Piomelli, U.: *Large-Eddy Simulation: Present State and Future Perspectives.* AIAA Paper 98-0534, 1998.

[208] Reynolds, O.: *On the Dynamical Theory of Incompressible Viscous Fluids and the Determination of the Criterion.* Phil. Trans. Roy. Soc., London, Series A 186 (1895), pp. 123-164.

[209] Schlichting, H.: *Boundary Layer Theory.* McGraw Hill, New York, 1968.

[210] Young, A.D.: *Boundary Layers.* BSP Professional Books, Blackwell Scientific Publication Ltd, Oxford, 1989.

[211] Favre, A.: *Equations des gaz turbulents compressibles, part 1: formes générales.* Journal de Mécanique, 4 (1965), pp. 361-390.

[212] Favre, A.: *Equations des gaz turbulents compressibles, part 2: méthode des vitesses moyennes; méthode des vitesses moyennes pondérées par la masse volumique.* Journal de Mécanique, 4 (1965), pp. 391-421.

[213] Morkovin, M.V.: *Effects of Compressibility on Turbulent Flow.* The Mechanics of Turbulence, Favre, A. (ed.), Gordon & Breach, New York, 1964.

[214] Rotta, J.: *Statistische Theorie nichthomogener Turbulenz I.* Zeitschrift für Physik, 129 (1951), pp. 547-572.

[215] Rodi, W.: *A New Algebraic Relation for Calculating the Reynolds Stresses.* ZAMM 56 (1976), pp. 219-221.

[216] Speziale, C.G.: *A Review of Reynolds Stress Models for Turbulent Shear Flows.* ICASE Report No. 95-15, 1995.

[217] Hallbäck, M.; Henningson, D.S.; Johansson, A.V.; Alfredsson, P.H. (eds.): *Turbulence and Transition Modelling.* ERCOFTAC Series, Vol. 2, Kluwer Academic Publishers, Dordrecht, The Netherlands, 1996.

[218] Boussinesq, J.: *Essai sur la théorie des eaux courantes.* Mem. Pres. Acad. Sci., XXIII, 46, Paris, 1877.

[219] Boussinesq, J.: *Theorie de l' écoulement tourbillonant et tumulteur des liquides dans les lits rectiligues.* Comptes Rendus de l' Acad. des Sciences, CXXII (1896), p. 1293.

[220] Spalart, P.; Shur, M.: *On the Sensitization of Turbulence Models to Rotation and Curvature.* Aerospace Science and Technology, 5 (1997), pp. 297-302.

[221] Shur, M.; Strelets, M.; Travin, A.; Spalart, P.: *Turbulence Modeling in Rotating and Curved Channels: Assessment of the Spalart-Shur Correction Term.* AIAA Paper 98-0325, 1998.

[222] Shih, T.H.; Zhu, J.; Liou, W.W.; Chen, K.-H.; Liu, N.-S.; Lumley, J.: *Modeling of Turbulent Swirling Flows.* Proc. 11th Symposium on Turbulent Shear Flows, Grenoble, France, 1997; also NASA TM-113112, 1997.

[223] Chen, K.-H.; Liu, N.-S.: *Evaluation of a Non-Linear Turbulence Model Using Mixed Volume Unstructured Grids.* AIAA Paper 98-0233, 1998.

[224] Abdel Gawad, A.F.; Abdel Latif, O.E.; Ragab, S.A.; Shabaka, I.M.: *Turbulent Flow and Heat Transfer in Rotating Non-Circular Ducts with Non-linear k-ε Model.* AIAA Paper 98-0326, 1998.

[225] Baldwin, B.S.; Lomax, H.J.: *Thin Layer Approximation and Algebraic Model for Separated Turbulent Flow.* AIAA Paper 78-257, 1978.

[226] Spalart, P.; Allmaras, S.: *A One-Equation Turbulence Model for Aerodynamic Flows.* AIAA Paper 92-0439, 1992.

[227] Launder, B.E.; Spalding, B.: *The Numerical Computation of Turbulent Flows.* Comput. Methods Appl. Mech. Eng. 3 (1974), pp. 269-289.

[228] Wilcox, D.C.: *Reassessment of the Scale Determining Equation for Advanced Turbulence Models.* AIAA Journal 26 (1988), pp. 1299-1310.

[229] Bardina, J.E.; Huang, P.G.; Coakley, T.J.: *Turbulence Modeling Validation, Testing, and Development.* NASA TM-110446, 1997.

[230] Jameson, A.: *A Non-Oscillatory Shock Capturing Scheme Using Flux Limited Dissipation.* MAE Report No. 1653, Dept. of Mechanical and Aerospace Engineering, Princeton University, 1984.

[231] Whitfield, D.L.; Janus, J.M.: *Three-Dimensional Unsteady Euler Equations Solution Using Flux Vector Splitting.* AIAA Paper 84-1552, 1984.

[232] Thomas, J.L.; Salas, M.D.: *Far Field Boundary Conditions for Transonic Lifting Solutions to the Euler Equations.* AIAA Journal, 24 (1986), pp. 1074-1080.

[233] Usab, W.J.; Murman, E.M.: *Embedded Mesh Solution of the Euler Equations Using a Multiple-Grid Method.* AIAA Paper 83-1946, 1983.

[234] Radespiel, R.: *A Cell-Vertex Multigrid Method for the Navier-Stokes Equations.* NASA TM-101557, 1989.

[235] Kroll, N.; Jain, R.K.: *Solution of Two-Dimensional Euler Equations – Experience with a Finite Volume Code.* DFVLR-FB 87-41, 1987.

[236] Verhoff, A.: *Modeling of Computational and Solid Surface Boundary Conditions for Fluid Dynamics Calculations.* AIAA Paper 85-1496, 1985.

[237] Hirsch, C.; Verhoff, A.: *Far Field Numerical Boundary Conditions for Internal and Cascade Flow Computations.* AIAA Paper 89-1943, 1989.

[238] Giles, M.B.: *Non-reflecting Boundary Conditions for Euler Equation Calculations.* AIAA Paper 89-1942, 1989.

[239] Tan, Z.; Wang, D.M.; Srininvasan, K.; Pzekwas, A.; Sun, R.: *Numerical Simulation of Coupled Radiation and Convection from Complex Geometries.* AIAA Paper 98-2677, 1998.

[240] Agarwal, R.K.; Schulte, P.: *A New Computational Algorithm for the Solution of Radiation Heat Transfer Problems.* AIAA Paper 98-2836, 1998.

[241] Hino, T.: *An Unstructured Grid Method for Incompressible Viscous Flows with a Free Surface.* AIAA Paper 97-0862, 1997.

[242] Löhner, R.; Yang, C.; Onate, E.; Idelsson, S.: *An Unstructured Grid-Based, Parallel Free Surface Solver.* AIAA Paper 97-1830, 1997.

[243] Cowles, G.; Martinelli, L.: *A Cell-Centered Parallel Multiblock Method for Viscous Incompressible Flows with a Free Surface.* AIAA Paper 97-1865, 1997.

[244] Wagner, C.A.; Davis, D.W.; Slimon, S.A.; Holligsworth, T.A.: *Computation of Free Surface Flows Using a Hybrid Multiblock/Chimera Approach.* AIAA Paper 98-0228, 1998.

Chapter 4

Spatial Discretisation: Structured Finite Volume Schemes

As we already mentioned in the introduction to Chapter 3, the majority of numerical schemes for the solution of the Euler- and the Navier-Stokes equations employ the *method of lines*, i.e., a separate discretisation in space and time. By consequence, it allows us to use numerical approximations of different accuracy for the spatial and temporal derivatives, as it may be required by the problem to be solved. Thus, we gain a lot of flexibility by this approach. For this reason, we shall follow the method of lines here. A detailed discussion of numerical methods based on *coupled* space and time discretisation, like the Lax-Wendroff family of schemes (e.g., explicit MacCormack predictor-corrector scheme, implicit Lerat's scheme, etc.), may be found, e.g., in Ref. [1].

A general, structured, finite volume scheme is naturally based on the conservation laws, which are expressed by the Navier-Stokes (2.19) or the Euler (2.45) equations. In a pre-processing step, the physical space is subdivided into a number of grid cells – quadrilaterals in 2D, hexahedra in 3D. The grid generation is done in such a way that:

- the domain is completely covered by the grid,
- there is no free space left between the grid cells,
- the grid cells do not overlap each other.

The resulting structured grid is uniquely described by the coordinates x, y, z of the grid points (corners of the grid cells) and indices in the computational space (see Fig. 3.2), let us call them i, j, k. Based on the grid, control volumes are defined in order to evaluate the integrals of the convective and viscous fluxes as well as of the source term. For simplicity, let us suppose that a particular

control volume does not change in time (otherwise please refer to Appendix A.4). Then, the time derivative of the conservative variables $\vec{W}$ can be cast in the form

$$\frac{\partial}{\partial t}\int_{\Omega}\vec{W}\,d\Omega = \Omega\,\frac{\partial\vec{W}}{\partial t}\,.$$

Herewith, Eq. (2.19) becomes

$$\frac{\partial\vec{W}}{\partial t} = -\frac{1}{\Omega}\left[\oint_{\partial\Omega}(\vec{F}_c - \vec{F}_v)\,dS - \int_{\Omega}\vec{Q}\,d\Omega\right]. \tag{4.1}$$

The surface integral on the right-hand side of Equation (4.1) is approximated by a sum of the fluxes crossing the faces of the control volume. This approximation is called *spatial discretisation*. It is usually supposed that the flux is constant along the individual face and that it is evaluated at the midpoint of the face. The source term is generally assumed to be constant inside the control volume. However, in cases where the source term becomes dominant, it is advisable to evaluate $\vec{Q}$ as the weighted sum of values from the neighbouring control volumes (see [2] and the references cited therein). If we consider a particular volume $\Omega_{I,J,K}$, as displayed in Fig. 4.1b, we obtain from Eq. (4.1)

$$\frac{d\vec{W}_{I,J,K}}{dt} = -\frac{1}{\Omega_{I,J,K}}\left[\sum_{m=1}^{N_F}(\vec{F}_c - \vec{F}_v)_m\,\Delta S_m \;-\; (\vec{Q}\Omega)_{I,J,K}\right]. \tag{4.2}$$

In the above expression, the indices in capital letters (I, J, K) reference the control volume, since in general it does not necessarily coincide with the grid, as we shall see later. Furthermore, N_F denotes the number of control volume faces (which is $N_F = 4$ in 2D and $N_F = 6$ in 3D). The variable ΔS_m stands for the area of the face m. The term in square brackets on the right-hand side of Eq. (4.2) is also generally termed the *residual*. It is denoted here by $\vec{R}_{I,J,K}$. Hence, we can abbreviate Eq. (4.2) as

$$\frac{d\vec{W}_{I,J,K}}{dt} = -\frac{1}{\Omega_{I,J,K}}\,\vec{R}_{I,J,K}\,. \tag{4.3}$$

When we write down the relationship in Equation (4.3) for all control volumes $\Omega_{I,J,K}$, we obtain a system of ordinary differential equations of first order. The equations are hyperbolic in time, that means we have to advance them in time starting from a known initial solution. We have also to provide suitable boundary conditions for the viscous and the inviscid fluxes, as they are described in Chapter 8.

When numerically solving the system of discretised governing equations (4.3), the first question is how to define the control volumes and where to locate the flow variables with respect to the computational grid. In the framework of structured finite volume schemes, three basic strategies are available:

- *Cell-centred* scheme – control volumes are identical with the grid cells and the flow variables are associated with their centroids (Fig. 4.3).

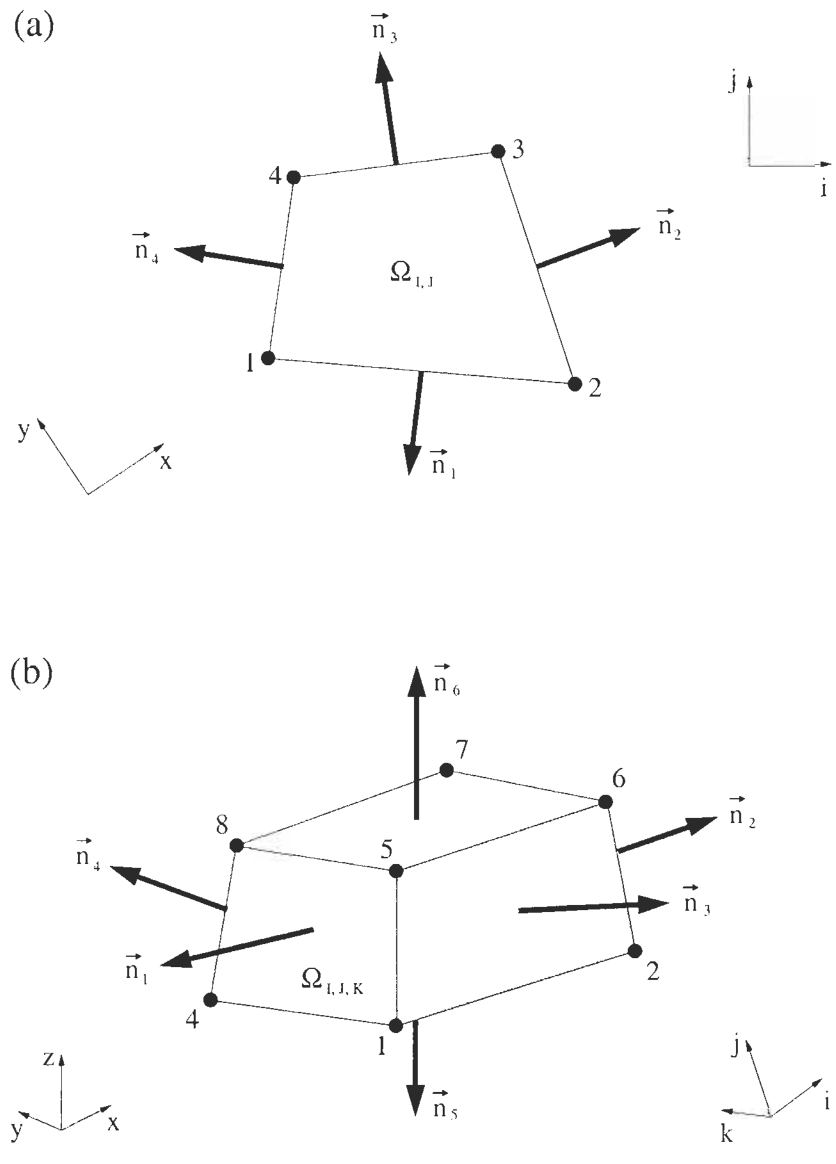


Figure 4.1: Control volume (Ω) and associated face unit normal vectors ($\vec{n}_m$) for a structured grid in: (a) two dimensions, (b) three dimensions. In case (a), the unit normal vectors $\vec{n}_2$ and $\vec{n}_4$ are associated with the i-coordinate (direction) in computational space, $\vec{n}_1$ and $\vec{n}_3$ with the j-coordinate. In case (b), the unit normal vectors $\vec{n}_1$ and $\vec{n}_2$ are associated with the i-coordinate, $\vec{n}_5$ and $\vec{n}_6$ with the j-coordinate, and $\vec{n}_3$, $\vec{n}_4$ with the k-coordinate, respectively.

- *Cell-vertex* scheme with *overlapping* control volumes – flow quantities are assigned to the grid points (vertices, nodes) and the control volumes are defined as the union of all grid cells having the respective vertex in common (4 cells in 2D, 8 cells in 3D – see Fig. 4.4). This means that the control volumes associated with two neighbouring grid points overlap each other.

- Cell-vertex scheme with *dual* control volumes – flow variables are again stored at the grid vertices, but the control volumes are now created by connecting the midpoints of the cells having the respective vertex in common (Fig. 4.5). In this way, the grid points are surrounded by their corresponding control volumes which do not overlap.

All three methodologies will be outlined in Section 4.2, which is devoted to general concepts of discretisation schemes. At this point, it should be mentioned that the cell-vertex scheme with overlapping control volumes is only seldom used today. Nevertheless, it is included here for completeness.

It is important to notice that in our case **all** flow variables, i.e., the conservative variables (ρ, ρu, ρv, ρw and ρE) and the dependent variables (p, T, c, etc.), are associated with the **same** location – with the cell centre or with the grid node. This approach is known as the *co-located* grid scheme. By contrast, many older pressure-based methods (cf. Section 3.1) use the so-called *staggered* grid scheme, where the pressure and the velocity components are stored at different locations in order to suppress oscillations of the solution which arise from central differencing.

A wide range of choices exists with respect to the evaluation of the convective fluxes. The basic problem is that we have to know their values at all N_F faces of a control volume, but the flow variables are not directly available there. This means, we have to interpolate either the fluxes or the flow variables to the face of the control volume. This can be in principle done in two ways:

- by arithmetic averaging like in *central* discretisation schemes;

- by some biased interpolation like in *upwind* discretisation schemes, which take care of the characteristics of the flow equations.

Besides the description, we shall treat aspects such as accuracy, range of applicability and numerical effort of the most widely used discretisation schemes for the convective fluxes in Section 4.3.

A commonly applied methodology for the evaluation of the viscous fluxes at a face of the control volume is based on arithmetic averaging of the flow quantities. More involved is the calculation of the velocity and the temperature gradients in Equations (2.15) and (2.24). We shall present the whole procedure in Section 4.4.

4.1 Geometrical Quantities of a Control Volume

Before we turn our attention to the discretisation methodologies, it is instructive to consider the calculation of geometrical quantities of the control volume $\Omega_{I,J,K}$ – its volume, the unit normal vector $\vec{n}_m$ (defined as outward facing) and the area ΔS_m of a face m. The normal vector and the face area are also denoted as the *metrics* of the control volume. In the following, we shall consider the 2-D and the 3-D case separately for a general quadrilateral or hexahedral control volume, respectively.

4.1.1 Two-Dimensional Case

In general, we consider flow in a plane as a special case of a 3-D problem, where the solution is symmetric with respect to one coordinate direction (e.g., to the z-direction). Because of the symmetry and in order to obtain correct physical units for volume, pressure, etc., we set the depth of all grid cells and control volumes equal to a constant value b. The volume of a control volume results then in two dimensions from the product of its area with the depth b. The area of a quadrilateral can be exactly calculated by the formula of Gauss. Hence, for a control volume like that displayed in Fig. 4.1a, we get after some algebra

$$\Omega_{I,J} = \frac{b}{2}\left[(x_1 - x_3)(y_2 - y_4) + (x_4 - x_2)(y_1 - y_3)\right]. \tag{4.4}$$

In the above, we have assumed that the control volume is located in the x-y-plane and that the z-coordinate is the symmetry axis. Since the depth b is arbitrary, we may set $b = 1$ for convenience. In two dimensions, the faces of a control volume are given by straight lines and therefore the unit normal vector is constant along them. When we integrate the fluxes according to the approximation of Eq. (4.2), we have to evaluate the product of the area of a face ΔS and the corresponding unit normal vector $\vec{n}$ – the *face vector* $\vec{S}$

$$\vec{S}_m = \begin{bmatrix} S_{x,m} \\ S_{y,m} \end{bmatrix} = \vec{n}_m \, \Delta S_m . \tag{4.5}$$

Because of the symmetry, the z-component of the face vectors (and the unit normal vector) is zero. It is therefore dropped from the expressions. The face vectors of the control volume from Fig. 4.1a are given by the relations

$$\begin{aligned}
\vec{S}_1 &= b\begin{bmatrix} y_2 - y_1 \\ x_1 - x_2 \end{bmatrix} \\
\vec{S}_2 &= b\begin{bmatrix} y_3 - y_2 \\ x_2 - x_3 \end{bmatrix} \\
\vec{S}_3 &= b\begin{bmatrix} y_4 - y_3 \\ x_3 - x_4 \end{bmatrix} \\
\vec{S}_4 &= b\begin{bmatrix} y_1 - y_4 \\ x_4 - x_1 \end{bmatrix} .
\end{aligned} \tag{4.6}$$

The unit normal vector at face m is then obtained from Eq. (4.5) as

$$\vec{n}_m = \frac{\vec{S}_m}{\Delta S_m} \tag{4.7}$$

with

$$\Delta S_m = |\ \vec{S}_m\ | = \sqrt{S^2_{x,m} + S^2_{y,m}}\ .$$

In practice, usually only the face vectors $\vec{S}_1$ and $\vec{S}_4$ are computed and stored for each control volume $\Omega_{I,J}$. The face vectors $\vec{S}_2$ as well as $\vec{S}_3$ are taken (with reversed signs to become outward facing) from the appropriate neighbouring control volumes in order to save memory and reduce the number of operations.

4.1.2 Three-Dimensional Case

As opposed to the previous 2-D case, the calculation of face vectors and volumes poses some problems in three dimensions. The main reason for this is that, in general, the four vertices of the face of a control volume may not lie in a plane. Then, the normal vector is no longer constant on the face (Fig. 4.2). In

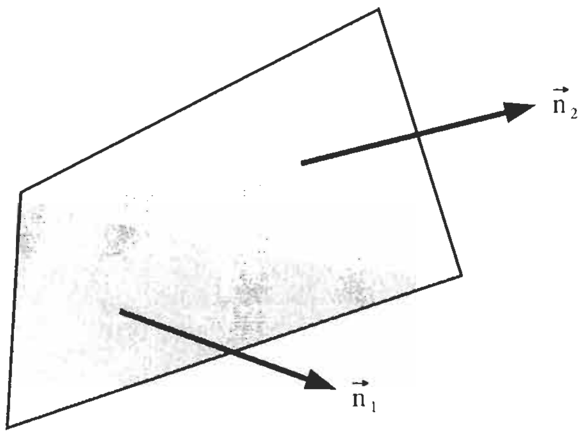


Figure 4.2: Face of a control volume with varying normal vector in 3D.

order to overcome this difficulty, we could decompose all six faces of the control volume into two or more triangles each. The volume itself could then be built of tetrahedra. Performing this subdivision in an appropriate manner would lead to a discretisation scheme which is at least first-order accurate on arbitrary grids [3]. Of course, the numerical effort would be increased substantially, because the fluxes would have to be integrated over each partial triangle separately. Hence,

the number of point operations would be at least doubled. However, in [3], [4] it is shown that for reasonably smooth grids, where the control volume faces approach parallelograms, the decomposition into triangles does not noticeably improve the solution accuracy. Therefore, we shall employ a simplified treatment of the quadrilateral faces in the following considerations, which is based on an averaged normal vector.

A face vector $\vec{S}$ of an hexahedral control volume, like that rendered in Fig. 4.1b, is most conveniently computed using the same Gauss's formula as employed in 2D for the area of a quadrilateral. Thus, e.g., for the face $m = 1$ (points 1, 5, 8 and 4 in Fig. 4.1b) we first define the differences

$$\begin{aligned} \Delta x_A &= x_8 - x_1 \,, \quad & \Delta x_B &= x_5 - x_4 \,, \\ \Delta y_A &= y_8 - y_1 \,, \quad & \Delta y_B &= y_5 - y_4 \,, \\ \Delta z_A &= z_8 - z_1 \,, \quad & \Delta z_B &= z_5 - z_4 \,. \end{aligned} \tag{4.8}$$

The face vector $\vec{S}_1 = \vec{n}_1 \Delta S_1$ results then from

$$\vec{S}_1 = \frac{1}{2} \begin{bmatrix} \Delta y_A \, \Delta z_B - \Delta z_A \, \Delta y_B \\ \Delta z_A \, \Delta x_B - \Delta x_A \, \Delta z_B \\ \Delta x_A \, \Delta y_B - \Delta y_A \, \Delta x_B \end{bmatrix} . \tag{4.9}$$

The five remaining face vectors are calculated in similar manner. It is again very convenient to store only three of the six the face vectors (e.g., $\vec{S}_1$, $\vec{S}_3$, and $\vec{S}_5$) for each control volume $\Omega_{I,J,K}$. The remaining face vectors $\vec{S}_2$, $\vec{S}_4$ as well as $\vec{S}_6$ are obtained (with reversed signs to become outward facing) from the appropriate neighbouring control volumes. The above expressions in Eq. (4.8) and (4.9) deliver an average face vector. The approximation becomes exact when the face approaches a parallelogram, i.e., when the vertices of the face lie all in one plane. The unit normal vector is obtained from Eq. (4.7) with

$$\Delta S_m = \sqrt{S_{x,m}^2 + S_{y,m}^2 + S_{z,m}^2} \,. \tag{4.10}$$

Various, more or less accurate formulae are available for the calculation of the volume of a general hexahedron (see, e.g., [1]). One approach, which performed very well in various applications, is based on the divergence theorem [5]. This relates the volume integral of the divergence of some vector quantity to its surface integral. The key idea is to use the location in space of some point of the control volume Ω, let us call it $\vec{r} = [r_x, r_y, r_z]^T$, as the vector quantity. Herewith, the divergence theorem reads

$$\int_\Omega \operatorname{div}(\vec{r}) \, d\Omega = \oint_{\partial\Omega} (\vec{r} \cdot \vec{n}) \, dS \,. \tag{4.11}$$

We can easily evaluate the left-hand side of Eq. (4.11) which gives us the volume

of Ω that we are looking for

$$\int_{\Omega} \operatorname{div}(\vec{r})\, d\Omega = \int_{\Omega} \left(\frac{\partial r_x}{\partial x} + \frac{\partial r_y}{\partial y} + \frac{\partial r_z}{\partial z} \right) d\Omega$$

$$= 3\,\Omega\,. \tag{4.12}$$

If we now assume constant unit normal vector on all faces of the control volume, we can solve the surface integral on the right-hand side of Eq. (4.11) as follows

$$\oint_{\partial\Omega} (\vec{r}\cdot\vec{n})\, dS \approx \sum_{m=1}^{m=6} (\vec{r}_{\mathrm{mid}}\cdot\vec{n})_m\, \Delta S_m\,. \tag{4.13}$$

In Eq. (4.13), $\vec{r}_{\mathrm{mid},m}$ denotes the midpoint of the control volume face m. For example,

$$\vec{r}_{\mathrm{mid},1} = \frac{1}{4}(\vec{r}_1 + \vec{r}_5 + \vec{r}_8 + \vec{r}_4)\,,$$

where the vectors $\vec{r}_1$, $\vec{r}_5$, $\vec{r}_8$, and $\vec{r}_4$ correspond to the vertices 1, 5, 8, and 4 of the face m=1 in Fig. 4.1b. Similar relations hold for the midpoints of the remaining faces. The area ΔS_m of the face m in Eq. (4.13) is obtained from Eq. (4.10). Combining Equations (4.12) and (4.13) together, and inserting the face vector $\vec{S}$ for the product $\vec{n}_m\,\Delta S_m$, we finally have the relationship

$$\Omega_{I,J,K} = \frac{1}{3}\sum_{m=1}^{m=6} (\vec{r}_{\mathrm{mid}}\cdot\vec{S})_m \tag{4.14}$$

for the volume of the control volume $\Omega_{I,J,K}$.

The origin of the coordinate system can be in principle moved to any place without affecting the volume calculation in Eq. (4.14). This leads us to the advice to locate the origin in one vertex of the control volume (e.g., point 1 in Fig. 4.1b), in order to achieve a better scaling of the numerical values. Thus, we may replace $\vec{r}$ in the above expressions (4.11)-(4.14) by a transformed vector $\vec{r}^{\,*}$, which is defined as

$$\vec{r}^{\,*} = \vec{r} - \vec{r}_{\mathrm{origin}}\,.$$

It is important to note that the volume calculated with aid of Eq. (4.14) is exact for a control volume with planar faces.

4.2 General Discretisation Methodologies

In the introduction to Chapter 4, we already mentioned the three approaches for the definition of the control volume and for the location of the flow variables. Here, we shall present all three in more detail. We shall also discuss their advantages and shortcomings.

4.2.1 Cell-Centred Scheme

We speak of a cell-centred scheme if the control volumes are identical with the grid cells and if the flow variables are located at the centroids of the grid cells as indicated in Fig. 4.3. When we evaluate the discretised flow equations (4.2), we have to supply the convective and the viscous fluxes at the faces of a cell [6]. They can be approximated in one of the three following ways:

1. by the *average of fluxes* computed from values at the centroids of the grid cells to the left and to the right of the cell face, but using the same face vector (generally applied only to the convective fluxes);

2. by using an *average of variables* associated with the centroids of the grid cells to the left and to the right of the cell face;

3. by computing the fluxes from flow quantities interpolated separately to the left and to the right side of the cell face (employed only for the convective fluxes).

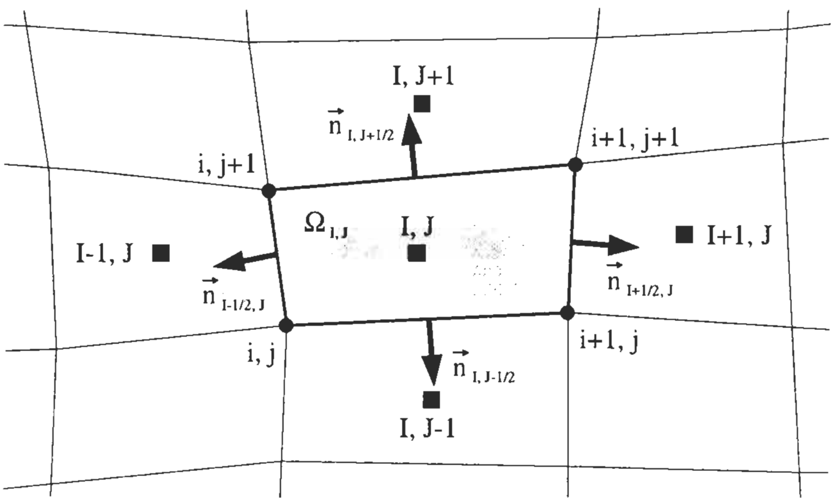


Figure 4.3: Control volume of a cell-centred scheme (in two dimensions).

Thus, taking the cell face $\vec{n}_{I+1/2,J}$ in Fig. 4.3 as an example, the first approach – average of fluxes – reads

$$\left(\vec{F}_c\,\Delta S\right)_{I+1/2,J} \approx \frac{1}{2}\left[\vec{F}_c(\vec{W}_{I,J}) + \vec{F}_c(\vec{W}_{I+1,J})\right]\,\Delta S_{I+1/2,J} \tag{4.15}$$

with $\Delta S_{I+1/2,J}$ computed from Eqs. (4.6) and (4.7).

The second possible approach – average of variables – can be formulated as follows

$$\left(\vec{F}\,\Delta S\right)_{I+1/2,J} \approx \vec{F}(\vec{W}_{I+1/2,J})\,\Delta S_{I+1/2,J}\,, \tag{4.16}$$

where the conservative/dependent variables at the face $\vec{n}_{I+1/2,J}$ of the control volume are defined as the arithmetic average of values at the two adjacent cells, i.e.,

$$\vec{W}_{I+1/2,J} = \frac{1}{2}\left(\vec{W}_{I,J} + \vec{W}_{I+1,J}\right)\,. \tag{4.17}$$

The flux vector $\vec{F}$ in Eq. (4.16) stands either for the convective or for the viscous fluxes.

The third methodology starts with an interpolation of flow quantities (being mostly velocity components, pressure, density and total enthalpy) separately to both sides of the cell face. The interpolated quantities – termed the *left* and the *right state* (see the begin of Section 4.3) – differ in general between both sides. The fluxes through the cell face are then evaluated from the difference of the left and right state using some non-linear function. Hence,

$$\left(\vec{F}_c\,\Delta S\right)_{I+1/2,J} \approx f_{Flux}\left(\vec{U}_L,\,\vec{U}_R,\,\Delta S_{I+1/2,J}\right)\,, \tag{4.18}$$

where

$$\begin{aligned}\vec{U}_L &= f_{Interp}\left(\cdots,\,\vec{U}_{I-1,J},\,\vec{U}_{I,J},\,\cdots\right)\\ \vec{U}_R &= f_{Interp}\left(\cdots,\,\vec{U}_{I,J},\,\vec{U}_{I+1,J},\,\cdots\right)\end{aligned} \tag{4.19}$$

represent the interpolated states. Of course, similar relations like (4.15)-(4.19) hold also for the other cell faces.

The same approximations are employed in three dimensions. For example, at the cell face $\vec{n}_{I+1/2,J,K}$ (e.g., identical to $\vec{n}_2$ in Fig. 4.1b) the average of fluxes in Eq. (4.15) becomes

$$\left(\vec{F}_c\,\Delta S\right)_{I+1/2,J,K} \approx \frac{1}{2}\left[\vec{F}_c(\vec{W}_{I,J,K}) + \vec{F}_c(\vec{W}_{I+1,J,K})\right]\,\Delta S_{I+1/2,J,K} \tag{4.20}$$

with $\Delta S_{I+1/2,J,K}$ being defined correspondingly to Equations (4.8) and (4.9). The average of variables reads similarly to Eq. (4.16) as

$$\left(\vec{F}\,\Delta S\right)_{I+1/2,J,K} \approx \vec{F}(\vec{W}_{I+1/2,J,K})\,\Delta S_{I+1/2,J,K} \tag{4.21}$$

with

$$\vec{W}_{I+1/2,J,K} = \frac{1}{2}\left(\vec{W}_{I,J,K} + \vec{W}_{I+1,J,K}\right)\,. \tag{4.22}$$

The way over the interpolation of the flow variables results similarly to Eq. (4.18) in

$$\left(\vec{F}_c \, \Delta S\right)_{I+1/2,J,K} \approx f_{Flux}\left(\vec{U}_L, \, \vec{U}_R, \, \Delta S_{I+1/2,J,K}\right), \tag{4.23}$$

where $\vec{U}_L$ and $\vec{U}_R$ are the interpolated values at the cell face.

The last term in the discretised flow equations (4.2) which remains to be evaluated is the source term $\vec{Q}$. As we already stated in the introduction, the source term is usually supposed to be constant inside the control volume. For this reason, it is calculated using the flow variables from the corresponding cell centre. Hence, we may define

$$(\vec{Q}\Omega)_{I,J,K} = \vec{Q}(\vec{W}_{I,J,K}) \, \Omega_{I,J,K} \,. \tag{4.24}$$

Using the above relations, the fluxes through the faces can be computed and the numerical integration over the boundary of $\Omega_{I,J,K}$ may be performed according to (4.2). In other words, the complete residual $\vec{R}_{I,J,K}$ is obtained. In Sections 4.3 and 4.4, we shall learn more about the details of the evaluation of the convective and viscous fluxes.

4.2.2 Cell-Vertex Scheme: Overlapping Control Volumes

In a cell-vertex scheme, all flow variables are associated with the nodes of the computational grid. Within the approach using overlapping control volumes, the grid cells still represent the control volumes, just as in the case of the cell-centred scheme. The difference is that now the residuals computed for the control volumes have to be distributed to the grid points [4], [7], [8]. The situation is sketched in Fig. 4.4.

Let us consider the control volume $\Omega_{I,J}$ in Fig. 4.4, which is defined by the nodes

$$(i,j) \qquad (i+1,j) \qquad (i+1,j+1) \qquad (i,j+1)\,.$$

Note that the point (i,j) is located at the lower left corner of $\Omega_{I,J}$. The convective fluxes, e.g., for the face $\Delta S_{I,J-1/2}$, which is given by the points (i,j) and $(i+1,j)$, are approximated as

$$\left(\vec{F}_c \, \Delta S\right)_{I,J-1/2} \approx \vec{F}_c(\vec{W}_{I,J-1/2}) \, \Delta S_{I,J-1/2} \,. \tag{4.25}$$

The variables at the midpoint of the face are evaluated using an arithmetic average of the variables at the nodes defining the face, i.e.,

$$\vec{W}_{I,J-1/2} = \frac{1}{2}\left[\vec{W}_{i,j} + \vec{W}_{i+1,j}\right]. \tag{4.26}$$

The face area $\Delta S_{I,J-1/2}$ is computed from the relations (4.6) and (4.7).

The approach remains the same in three dimensions. For example, for the face associated with the normal vector $\vec{n}_3$ in Fig. 4.1b, the averaged variables read

$$\vec{W}_{I,J,K-1/2} = \frac{1}{4}\left[\vec{W}_1 + \vec{W}_2 + \vec{W}_5 + \vec{W}_6\right]. \tag{4.27}$$

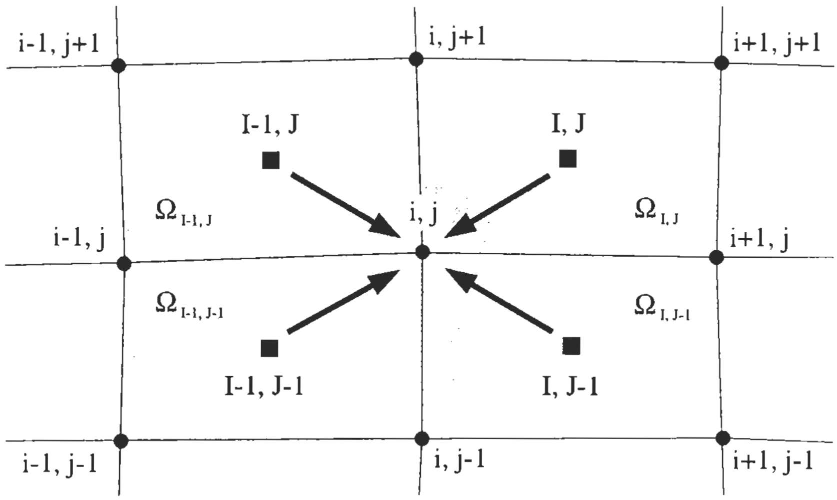


Figure 4.4: Overlapping control volumes of a cell-vertex scheme (2-D); arrows represent distribution of the residuals from cell centres to the common node i, j.

If we assume the edge 1-2 being oriented in the i-direction, edge 1-5 in the j-direction, edge 1-4 in the k-direction, and if we finally associate the point (i, j, k) with the corner 1 in Fig. 4.1b, the average in Eq. (4.27) can also be written as

$$\vec{W}_{I,J,K-1/2} = \frac{1}{4}\left[\vec{W}_{i,j,k} + \vec{W}_{i+1,j,k} + \vec{W}_{i,j+1,k} + \vec{W}_{i+1,j+1,k}\right] . \tag{4.28}$$

The convective flux is then again obtained from

$$\left(\vec{F}_c\, \Delta S\right)_{I,J,K-1/2} \approx \vec{F}_c(\vec{W}_{I,J,K-1/2})\, \Delta S_{I,J,K-1/2} \, , \tag{4.29}$$

where the face area $\Delta S_{I,J,K-1/2}$ results from the formulae (4.8) and (4.9). The viscous fluxes are normally computed employing the same approach as for the dual control-volume scheme [9], [10], which results in a more compact scheme (i.e., one which involves fewer nodal values).

Summing up all face contributions given by relations (4.25), (4.26) and (4.28), (4.29), respectively, we obtain the intermediate residuals $\vec{R}_{I,J,K}$ for all grid cells. In order to relate the cell-based to the node-based residuals, a further approximation is made using a *residual distribution formula*. It is basically a function, which evaluates the unknown node-based residual from a weighted sum of all cells having the particular grid node in common. The following distribution formulae were devised:

- volume weighted sum due to Ni [7];
- non-weighted sum due to Hall [8];
- characteristic (upwind) weighting procedure of Rossow [11], [12].

Theoretical investigations of the truncation error [4] suggest that Ni's scheme is more accurate than Hall's approach. However, in practice Ni's distribution formula leads to problems in places, where the grid is strongly distorted and stretched. For example, strong oscillations of the pressure field near the trailing edge of an airfoil were observed when using O-grids [13], [4]. Furthermore, convergence could only be achieved when the numerical viscosity was increased considerably. The underlying idea of the upwind weighting procedure [11], [12] is quite similar to that of the fluctuation-splitting schemes [14]-[17] (cf. Subsection 3.1.5), but the implementation is numerically much simpler. Basically, the residuals are sent only upstream in the characteristic direction.

Of the three approaches, Hall's distribution scheme proved to be the most robust. In Hall's scheme, the residual at a particular node results from a simple sum of all intermediate residuals $\vec{R}_{I,J,K}$, which cells share the node. Thus, in the 2-D case rendered in Fig. 4.4, we get

$$\vec{R}_{i,j} = \vec{R}_{I,J} + \vec{R}_{I-1,J} + \vec{R}_{I-1,J-1} + \vec{R}_{I,J-1} \,. \tag{4.30}$$

In three dimensions, in total eight cell-based residuals have to be summed up in the same way. A close inspection of (4.30) reveals that $\vec{R}_{i,j}$ is just the net flux through the boundary of the supercell

$$\Omega_{i,j} = \Omega_{I,J} + \Omega_{I-1,J} + \Omega_{I-1,J-1} + \Omega_{I,J-1} \,, \tag{4.31}$$

due to the fact that the fluxes across the inner faces cancel each other. The supercell also represents the "total" control volume, centred at the point (i,j). In the 3-D case, the total volume consists of the cells

$$\begin{aligned} \Omega_{i,j,k} = &\, \Omega_{I,J,K} + \Omega_{I-1,J,K} + \Omega_{I-1,J-1,K} + \Omega_{I,J-1,K} \\ &+ \Omega_{I,J,K-1} + \Omega_{I-1,J,K-1} + \Omega_{I-1,J-1,K-1} + \Omega_{I,J-1,K-1} \,. \end{aligned} \tag{4.32}$$

As it can be seen from Fig. 4.4, the control volumes overlap by at least one cell, which gave the scheme its name.

The source term is calculated using the flow variables from the corresponding grid node, i.e.,

$$(\vec{Q}\Omega)_{i,j,k} = \vec{Q}(\vec{W}_{i,j,k})\, \Omega_{i,j,k} \,. \tag{4.33}$$

With the above definitions, the time-stepping scheme in Eq. (4.2) becomes

$$\frac{d\vec{W}_{i,j,k}}{dt} = -\frac{1}{\Omega_{i,j,k}} \vec{R}_{i,j,k} \,. \tag{4.34}$$

This represents a system of ordinary differential equations, which has to be solved in each grid point (i,j,k) in the same way as for the cell-centred scheme. Note that the total volume (4.31) or (4.32), respectively, is utilised in Eq. (4.34).

4.2.3 Cell-Vertex Scheme: Dual Control Volumes

In this scheme, the control volumes are centred around the particular grid node (vertex), where all flow variables are stored [18], [19]. As depicted in Fig. 4.5, in the 2-D case the dual control volumes are constructed by joining the midpoints of the four cells which share the node. In three dimensions, the centroids of eight cells have to be connected in order to form the faces of the control volume. Another possibility would be to join one cell centroid to the edge midpoint (in two dimensions) and then to the neighbouring cell centroid again [20]. Thus, the face of the control volume would consist of two parts with different normal vectors, as it is common for unstructured grids (see next Chapter). However, in the case of structured grids, such a definition of the control volume is justified only at boundaries, where the surface discretisation would be otherwise changed. This is demonstrated in Fig. 4.6. Significant advantages with respect to accuracy in the interior field cannot be expected from the second approach on reasonably smooth grids. Therefore, in what follows, the simpler definition of the dual control volume will be employed. The boundary treatment is discussed in Chapter 8.

When we evaluate the discretised flow Equations (4.2), we have to compute the convective and the viscous fluxes at the faces of the control volume. This can be done according to one of the following three approaches:

1. by the *average of fluxes* computed from values at the nodes to the left and to the right of the face of the control volume, but using the same face vector (generally applied only to the convective fluxes);

2. by using an *average of variables* stored at the nodes to the left and to the right of the face;

3. by computing the fluxes from flow quantities interpolated separately to the left and to the right side of the face (employed only for the convective fluxes).

Thus, for example at the cell face $\vec{n}_{i+1/2,j}$ in Fig. 4.5 the first approach – average of fluxes – reads

$$\left(\vec{F}_c\,\Delta S\right)_{i+1/2,j} \approx \frac{1}{2}\left[\vec{F}_c(\vec{W}_{i,j}) + \vec{F}_c(\vec{W}_{i+1,j})\right]\Delta S_{i+1/2,j} \tag{4.35}$$

with $\Delta S_{i+1/2,j}$ being computed according to Eq. (4.6) and (4.7), respectively, from the known coordinates of the cell centroids.

The second possible approach – average of variables – can be formulated as follows

$$\left(\vec{F}\,\Delta S\right)_{i+1/2,j} \approx \vec{F}(\vec{W}_{i+1/2,j})\,\Delta S_{i+1/2,j}\,, \tag{4.36}$$

where the conservative (or the dependent) variables at the face $\vec{n}_{i+1/2,j}$ of the control volume result from arithmetic averaging of values at the two neighbouring nodes. Hence,

$$\vec{W}_{i+1/2,j} = \frac{1}{2}\left(\vec{W}_{i,j} + \vec{W}_{i+1,j}\right)\,. \tag{4.37}$$

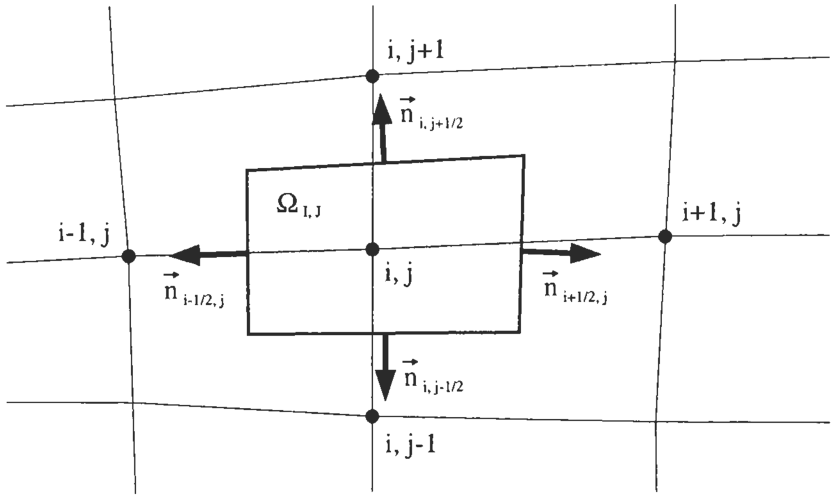


Figure 4.5: Dual control volume of a cell-vertex scheme in two dimensions.

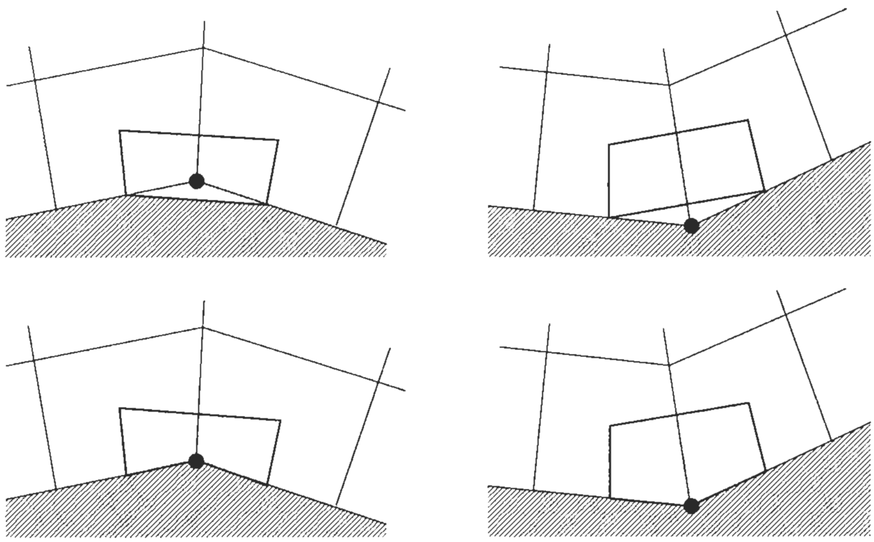


Figure 4.6: Definition of the dual control volume at a boundary (2D). Upper part: by connecting edge midpoints. Lower part: by connecting edge midpoints and the central node at the boundary.

The flux vector $\vec{F}$ in Eq. (4.36) represents either the convective or the viscous fluxes.

The third methodology utilises an interpolation of flow quantities (being mostly velocity components, pressure, density and total enthalpy) separately to both sides of the face. The interpolated quantities – termed the *left* and the *right state* (see the begin of Section 4.3) – differ in general between both sides. The fluxes through the face of the control volume are then evaluated from the difference of the left and right state using some non-linear function. Thus,

$$\left(\vec{F}_c\,\Delta S\right)_{i+1/2,j} \approx f_{Flux}\left(\vec{U}_L,\,\vec{U}_R,\,\Delta S_{i+1/2,j}\right)\,, \tag{4.38}$$

where

$$\begin{aligned}\vec{U}_L &= f_{Interp}\left(\cdots,\,\vec{U}_{i-1,j},\,\vec{U}_{i,j},\,\cdots\right)\\ \vec{U}_R &= f_{Interp}\left(\cdots,\,\vec{U}_{i,j},\,\vec{U}_{i+1,j},\,\cdots\right)\end{aligned} \tag{4.39}$$

stand for the interpolated states. Of course, similar relations like (4.35)-(4.39) apply in the same way to other faces of the control volume.

The same approximations are employed in three dimensions. For example, at the cell face $\vec{n}_{i+1/2,j,k}$ (e.g., identical to $\vec{n}_2$ in Fig. 4.1b) the average of fluxes in Eq. (4.35) becomes

$$\left(\vec{F}_c\,\Delta S\right)_{i+1/2,j,k} \approx \frac{1}{2}\left[\vec{F}_c(\vec{W}_{i,j,k}) + \vec{F}_c(\vec{W}_{i+1,j,k})\right]\Delta S_{i+1/2,j,k}\,, \tag{4.40}$$

where $\Delta S_{i+1/2,j,k}$ is obtained from the Equations (4.8) and (4.9). The average of the flow variables results similarly to Eq. (4.36) in

$$\left(\vec{F}\,\Delta S\right)_{i+1/2,j,k} \approx \vec{F}(\vec{W}_{i+1/2,j,k})\,\Delta S_{i+1/2,j,k} \tag{4.41}$$

with

$$\vec{W}_{i+1/2,j,k} = \frac{1}{2}\left(\vec{W}_{i,j,k} + \vec{W}_{i+1,j,k}\right)\,. \tag{4.42}$$

Finally, the interpolation of the flow variables leads correspondingly to Eq. (4.38) to

$$\left(\vec{F}_c\,\Delta S\right)_{i+1/2,j,k} \approx f_{Flux}\left(\vec{U}_L,\,\vec{U}_R,\,\Delta S_{i+1/2,j,k}\right)\,, \tag{4.43}$$

where $\vec{U}^L$ and $\vec{U}^R$ denote the interpolated values at the face. A detailed description of several possible approaches will be presented in Section 4.3 for the convective and in Section 4.4 for the viscous fluxes.

The last term in the discretised flow Equations (4.2) to be evaluated is the source term $\vec{Q}$. As already stated in the introduction, the source term is supposed to be constant inside the control volume. For this reason, it is computed using the flow variables from the corresponding grid point. Hence, we may define

$$(\vec{Q}\Omega)_{i,j,k} = \vec{Q}(\vec{W}_{i,j,k})\;\Omega_{i,j,k}\,. \tag{4.44}$$

Using the above relations, the fluxes through the faces can be computed and the numerical integration over the boundary of $\Omega_{i,j,k}$ can be carried out according to Eq. (4.2). In this way, we obtain the complete residual $\vec{R}_{i,j,k}$ including the source term. The change in time of the conservative variables follows then for each grid point from

$$\frac{d\vec{W}_{i,j,k}}{dt} = -\frac{1}{\Omega_{i,j,k}}\,\vec{R}_{i,j,k}\,. \tag{4.45}$$

Suitable solution methods will be presented later in Chapter 6.

4.2.4 Cell-Centred versus Cell-Vertex Schemes

In the preceding three sections, both the cell-centred and the cell-vertex discretisation methodologies were outlined. The following paragraphs compare the three schemes and give an overview of the at times controversial debate about their relative merits.

First, let us consider the accuracy of the discretisations. It follows from the discussion in [4], [21] that the cell-vertex scheme (either with overlapping or dual control volumes) can be made first-order accurate on distorted grids. On Cartesian or on smooth grids (i.e., where the volumes between adjacent cells vary only moderately and which are only slightly skewed), the cell-vertex scheme is second- or higher-order accurate [22], depending on the flux evaluation scheme. In the opposite, the discretisation error of a cell-centred scheme depends strongly on the smoothness of the grid. For example, for an arrangement of the cells sketched in Fig. 4.7, an averaging does not provide the correct value at the midpoint of a face even for a linearly varying function. The consequence is that on a grid with slope discontinuity the discretisation error will not be reduced even when the grid is infinitely refined. As demonstrated in [4], such zero-order errors manifest themselves as oscillations or kinks in isolines, whereas a cell-vertex scheme experiences no problems in the same situation. Nevertheless, on Cartesian or on sufficiently smooth grids, the cell-centred scheme can also reach second- or higher-order accuracy. A further analysis of the discretisation errors were presented in [23]-[26].

Second, let us compare the three methods and their characteristics at boundaries. It is mainly at the solid wall boundary where the cell-vertex scheme with dual control volumes faces difficulties. Recalling again Fig. 4.6, it is apparent that only about one half of the control volume is left at the boundary. The integration of fluxes around the faces results in a residual located **inside** – ideally in the **centroid** – of the control volume. But, the residual is associated with the **node** residing directly at the wall. This mismatch leads to increased discretisation error in comparison to the cell-centred scheme. The definition of the dual control volume causes also problems at sharp corners (like trailing edges), which show up as unphysical peaks in pressure or density. Further complications arise, e.g., at coordinate cuts or at periodic boundaries (see Chapter 8), where the fluxes from both parts of the control volume have to be summed

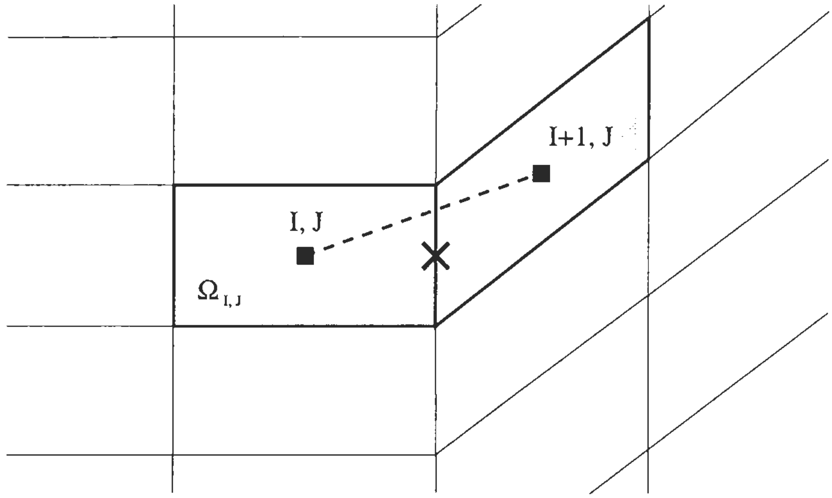


Figure 4.7: Cell-centred flux balance on a skewed grid; cross denotes the mid-point of the cell face.

up correctly. All cell-vertex schemes also require additional logic, in order to assure a consistent solution at boundary points shared by multiple blocks. No such problems appear for cell-centred schemes.

The cell-vertex scheme with overlapping control volumes has an advantage over the dual volume scheme in the treatment of wall boundaries, but it cannot be combined with the popular upwind discretisation methods like TVD, AUSM, or CUSP. The discretisation involves more points than those of the cell-centred and the dual control-volume schemes (27 instead of 7 in 3D), which leads to smearing of discontinuities and memory overhead in the case of an implicit time discretisation.

The last main difference between the cell-centred and the cell-vertex schemes appears for unsteady flow problems. As mentioned earlier in Section 3.2, the cell-vertex schemes require at least an approximate treatment of the mass matrix [27], [28]. On contrary, the mass matrix can be completely discarded in the case of a cell-centred scheme, because the residual is naturally associated with the centroid of the control volume.

In summary, the cell-vertex scheme with dual control volumes and the cell-centred scheme are numerically very similar in the interior of a stationary flow field. The main differences occur in the boundary treatment and for unsteady flows. In both cases, the cell-centred approach shows advantages over the cell-vertex schemes, which result in a more straightforward implementation in a flow solver.

4.3 Discretisation of Convective Fluxes

In the previous sections, we discussed the spatial discretisation methodologies in general. In this part, we shall learn more about the details, how the convective fluxes can be approximated.

As we could already see in Subsection 3.1.5, in the framework of the finite volume approach, we have basically the choice between:

- central,
- flux-vector splitting,
- flux-difference splitting,
- total variation diminishing (TVD), and
- fluctuation-splitting

schemes. In order to keep the amount of material bounded, we will restrict ourselves to the most important and popular methods. We will omit any detailed description of all possible modifications to the basic schemes, but instead reference the relevant literature.

Before we start to present the various discretisation schemes in detail, we should explain what is meant by the designations *left* and *right state*, as well as by *stencil* or *computational molecule*, respectively.

Certain cell-centred and dual control-volume schemes require an interpolation of flow variables to the faces of the control volume. The situation is sketched in Fig. 4.8 for a grid in the i-direction. One possibility, which is employed by the

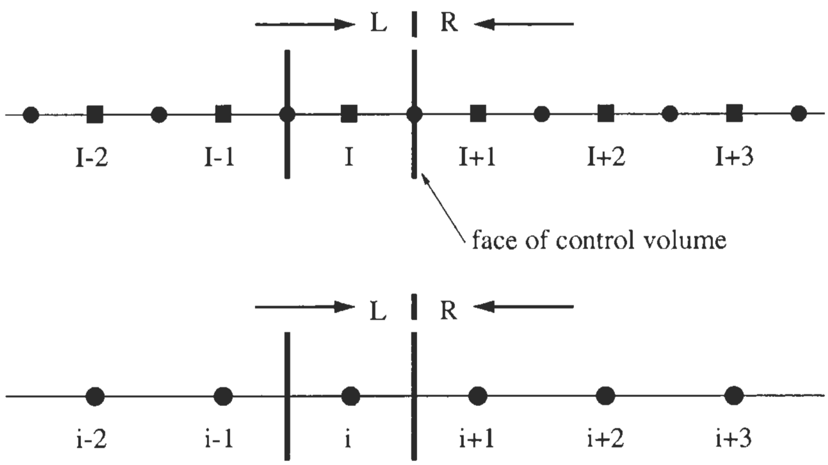


Figure 4.8: Left and right state at cell face $I+1/2$, resp. $i+1/2$. Upper part: cell-centred scheme; lower part: cell-vertex scheme with dual control volumes. Circles denote nodes, rectangles represent cell-centroids.

central scheme (see next subsection), consists of linear interpolation using the same number of values to the left and to the right of the face. In other words, the interpolation is centred at the face. Discretisations based on the characteristics of the Euler equations – upwind schemes – separately interpolate flow variables from the left and the right side of the face using non-symmetric formulae. The two values, named the **left** and the **right state**, are then utilised to compute the convective flux through the face (see Eqs. (4.18), (4.23), (4.38), or (4.43)). The interpolation formulae are almost exclusively (with the exception of the TVD schemes) based on Van Leer's MUSCL (Monotone Upstream-Centred Schemes for Conservation Laws) approach [29]. They read for a general flow variable U

$$\begin{aligned} U_R &= U_{I+1} - \frac{\epsilon}{4}\left[(1+\hat{\kappa})\Delta_- + (1-\hat{\kappa})\Delta_+\right] U_{I+1} \\ U_L &= U_I \quad + \frac{\epsilon}{4}\left[(1+\hat{\kappa})\Delta_+ + (1-\hat{\kappa})\Delta_-\right] U_I \,. \end{aligned} \tag{4.46}$$

The forward (Δ_+) and the backward (Δ_-) difference operators are defined as

$$\begin{aligned} \Delta_+ U_I &= U_{I+1} - U_I \\ \Delta_- U_I &= U_I - U_{I-1} \,. \end{aligned} \tag{4.47}$$

The indices are shifted as appropriate. The above relationships remain valid for a cell-vertex scheme with dual control volumes, if the node index i is substituted for I. The parameter ϵ can be set equal to zero to obtain a first-order accurate upwind discretisation. The parameter $\hat{\kappa}$ determines the spatial accuracy of the interpolation. For $\epsilon = 1$ and $\hat{\kappa} = -1$, the above interpolation formulae (4.46) give a fully one-sided interpolation of the flow variables, which results in a second-order accurate upwind approximation on uniformly spaced grid. The case $\hat{\kappa} = 0$ corresponds to a second-order accurate, upwind-biased linear interpolation. Furthermore, by setting $\hat{\kappa} = 1/3$, we obtain a three-point interpolation formula which constitutes (in a finite volume framework – cf. Ref. [30], [48]) a second-order upwind-biased scheme with lower truncation error than the $\hat{\kappa} = -1$ and $\hat{\kappa} = 0$ schemes. Finally, if we specify $\hat{\kappa} = 1$, the MUSCL approach reduces to a purely central scheme – the average of variables. The schemes with $\hat{\kappa} = 0$ and $\hat{\kappa} = 1/3$ are most often used in practice.

The MUSCL interpolation (4.46) has to be enhanced by the so-called *limiter function* or *limiter*, if the flow region contains strong gradients. The purpose of the limiter is to suppress non-physical oscillation of the solution. Limiters will be discussed further in Subsection 4.3.5.

Stencil or **computational molecule** stands for the union of those cell-centroids or grid points, which are involved in the computation of the residual, the gradient, etc. For example, if we average the fluxes at the faces of the control volume according to the Equations (4.15), (4.20), or (4.35), (4.40), respectively, we obtain in two dimensions a 5-point stencil, consisting of the cells/points

$$(I,J) \quad (I+1,J) \quad (I,J+1) \quad (I-1,J) \quad (I,J-1)\,.$$

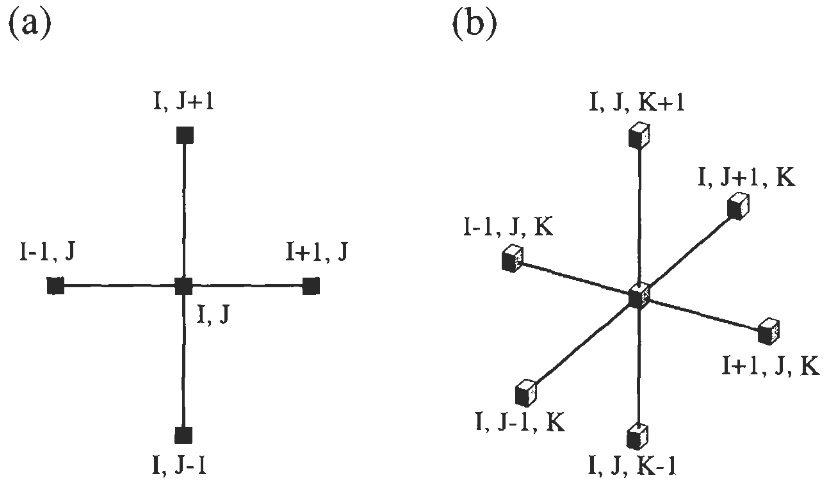


Figure 4.9: Stencil (computational molecule) of the central discretisation scheme; (a) in 2-D, (b) in 3-D space.

In three dimensions, a 7-point stencil results, which involves the cells/points

$$\begin{gathered}(I,J,K) \quad (I+1,J,K) \quad (I,J+1,K) \quad (I-1,J,K) \\ (I,J-1,K) \quad (I,J,K-1) \quad (I,J,K+1)\,.\end{gathered}$$

Both stencils are displayed in Fig. 4.9. Note that on a Cartesian grid this corresponds to the second-order accurate central-difference approximation of the first derivatives in i-, j-, and k-direction. Thus, a finite difference scheme, applied to the differential form of the governing equations and using the central differences, would deliver the same result.

4.3.1 Central Scheme with Artificial Dissipation

The central scheme with artificial dissipation is very simple compared to other discretisation methods. It is easy to implement with either the cell-centred scheme or with both cell-vertex schemes. For these reasons the scheme became very wide-spread.

The basic idea of the central scheme is to compute the convective fluxes at a face of the control volume from the arithmetic average of the conservative variables on both sides of the face. Since this would allow for odd-even decoupling of the solution (generation of two independent solutions of the discretised equations) and overshoots at shocks, artificial dissipation (which is similar to the viscous fluxes) has to be added for stability. The scheme was first implemented for the Euler equations by Jameson et al. [6]. Because of the names of the authors, it is also abbreviated as the JST scheme.

The central scheme is generally less accurate in the resolution of discontinuities and boundary layers than, say, the upwind schemes. However, it is computationally considerably cheaper. Therefore, attempts were made to improve the accuracy of the scheme, while still keeping the numerical effort low. For example, improvements were devised to reduce the amount of the artificial dissipation in boundary layers [31], [32], or to enhance the shock resolution [33]-[35]. Another idea, followed in [35], is to utilise the Jacobian matrix of the convective fluxes, in order to scale the dissipation independently for each conservation equation. This successful approach is known as the matrix dissipation scheme. It can be viewed as a compromise between the original scalar scheme and the upwind schemes. The basic scheme also employs a single, pressure-based sensor to switch from second- to first-order accuracy at discontinuities to prevent non-physical oscillations of the flow variables. In [36], Jameson developed a concept called the SLIP (Symmetric Limited Positive) scheme, where a limiter is applied separately for each conservation equation. A brief description of the previous approaches and comparisons for inviscid and viscous 2-D flows was presented in [37].

Scalar Dissipation Scheme

The convective fluxes (Eq. (2.21)) through a face of the control volume are approximated using the average of variables, according to the Equations (4.16), (4.21), (4.36), or (4.41), respectively. Artificial dissipation is then added to the central fluxes for stability [6], [38]. Thus, the total convective flux at face $(I+1/2, J, K)$ reads

$$\left(\vec{F}_c\,\Delta S\right)_{I+1/2,J,K} \approx \vec{F}_c(\vec{W}_{I+1/2,J,K})\,\Delta S_{I+1/2,J,K} - \vec{D}_{I+1/2,J,K}\,, \qquad (4.48)$$

where the flow variables are averaged as (see also Fig. 4.8)

$$\vec{W}_{I+1/2,J,K} = \frac{1}{2}\left(\vec{W}_{I,J,K} + \vec{W}_{I+1,J,K}\right). \qquad (4.49)$$

In the case of the cell-vertex scheme with dual control volumes, node indices (i, j, k) would be used instead. For simplicity, $(I+1/2, J, K)$ will be abbreviated as $(I+1/2)$ hereafter. The artificial dissipation flux consists of a blend of adaptive second- and fourth-order differences which result from the sum of first- and third-order difference operators

$$\begin{aligned}\vec{D}_{I+1/2} = \hat{\Lambda}^S_{I+1/2}\Big[&\epsilon^{(2)}_{I+1/2}\,(\vec{W}_{I+1} - \vec{W}_I)\\ &- \epsilon^{(4)}_{I+1/2}\,(\vec{W}_{I+2} - 3\vec{W}_{I+1} + 3\vec{W}_I - \vec{W}_{I-1})\Big]\,.\end{aligned} \qquad (4.50)$$

From Eq. (4.50) we can see that the scheme possesses a compact 9-point stencil in two dimensions and a 13-point stencil in three dimensions.

The dissipation is scaled by the sum of the spectral radii of the convective flux Jacobians in all coordinate directions

$$\hat{\Lambda}^S_{I+1/2} = (\hat{\Lambda}^I_c)_{I+1/2} + (\hat{\Lambda}^J_c)_{I+1/2} + (\hat{\Lambda}^K_c)_{I+1/2}\,. \qquad (4.51)$$

The spectral radius at the cell face $(I + 1/2)$, e.g., in I-direction (represented by the superscript I), results from the average

$$(\hat{\Lambda}_c^I)_{I+1/2} = \frac{1}{2}\left((\hat{\Lambda}_c^I)_I + (\hat{\Lambda}_c^I)_{I+1}\right) . \tag{4.52}$$

It is evaluated using the formula

$$\hat{\Lambda}_c = (\,|V| + c)\,\Delta S\,, \tag{4.53}$$

where V stands for the contravariant velocity (2.22) and c for the speed of sound, respectively.

A pressure-based sensor is used to switch off the fourth-order differences at shocks, where they would lead to strong oscillation of the solution. The sensor also switches off the second-order differences in smooth parts of the flow field, in order to reduce the dissipation to the lowest possible level. Herewith, the coefficients $\epsilon^{(2)}$ and $\epsilon^{(4)}$ in Eq. (4.50) are defined as

$$\begin{aligned}
\epsilon^{(2)}_{I+1/2} &= k^{(2)}\max(\Upsilon_I, \Upsilon_{I+1}) \\
\epsilon^{(4)}_{I+1/2} &= \max\left[0,\, (k^{(4)} - \epsilon^{(2)}_{I+1/2})\right]
\end{aligned} \tag{4.54}$$

with the pressure sensor given by

$$\Upsilon_I = \frac{|p_{I+1} - 2p_I + p_{I-1}|}{p_{I+1} + 2p_I + p_{I-1}} . \tag{4.55}$$

Typical values of the parameters are $k^{(2)} = 1/2$ and $1/128 \le k^{(4)} \le 1/64$.

In order to reduce the amount of artificial dissipation across a viscous shear layer, we can re-define the scaling factors in Eq. (4.50) as follows [31], [32]

$$\begin{aligned}
\hat{\Lambda}^S_{I+1/2} &= \frac{1}{2}\left[(\phi^I\hat{\Lambda}^I_c)_I + (\phi^I\hat{\Lambda}^I_c)_{I+1}\right] \\
\hat{\Lambda}^S_{J+1/2} &= \frac{1}{2}\left[(\phi^J\hat{\Lambda}^J_c)_J + (\phi^J\hat{\Lambda}^J_c)_{J+1}\right] \\
\hat{\Lambda}^S_{K+1/2} &= \frac{1}{2}\left[(\phi^K\hat{\Lambda}^K_c)_K + (\phi^K\hat{\Lambda}^K_c)_{K+1}\right] .
\end{aligned} \tag{4.56}$$

These are then employed instead of Eq. (4.51). The directionally dependent coefficients ϕ are given by the relations

$$\begin{aligned}
\phi^I &= 1 + \max\left[\left(\frac{\hat{\Lambda}^J_c}{\hat{\Lambda}^I_c}\right)^\sigma, \left(\frac{\hat{\Lambda}^K_c}{\hat{\Lambda}^I_c}\right)^\sigma\right] \\
\phi^J &= 1 + \max\left[\left(\frac{\hat{\Lambda}^I_c}{\hat{\Lambda}^J_c}\right)^\sigma, \left(\frac{\hat{\Lambda}^K_c}{\hat{\Lambda}^J_c}\right)^\sigma\right] \\
\phi^K &= 1 + \max\left[\left(\frac{\hat{\Lambda}^I_c}{\hat{\Lambda}^K_c}\right)^\sigma, \left(\frac{\hat{\Lambda}^J_c}{\hat{\Lambda}^K_c}\right)^\sigma\right] .
\end{aligned} \tag{4.57}$$

The parameter σ is usually set equal to 1/2 or 2/3. This formulation decreases the scaling of the dissipation terms in the direction along the shorter side of a control volume, whose longer side is aligned with the flow.

Matrix Dissipation Scheme

In order to improve the accuracy by reducing the numerical dissipation, the above scheme can be modified to become more like an upwind scheme. The idea is to use a matrix – the convective flux Jacobian – instead of the scalar value $\hat{\Lambda}^S$ to scale the dissipation terms [35]. In this way, each equation is scaled properly by the corresponding eigenvalue. Hence, the Eq. (4.50) becomes

$$\vec{D}_{I+1/2} = |\bar{A}_c|_{I+1/2} \Big[\epsilon^{(2)}_{I+1/2} (\vec{W}_{I+1} - \vec{W}_I) - \epsilon^{(4)}_{I+1/2} (\vec{W}_{I+2} - 3\vec{W}_{I+1} + 3\vec{W}_I - \vec{W}_{I-1}) \Big] . \tag{4.58}$$

The scaling matrix corresponds to the convective flux Jacobian ($\bar{A}_c = \partial \vec{F}_c / \partial \vec{W}$) diagonalised with absolute values of the eigenvalues

$$|\bar{A}_c| = \bar{T}\, |\bar{\Lambda}_c\, \Delta S|\, \bar{T}^{-1} . \tag{4.59}$$

The matrices of right ($\bar{T}$) and left ($\bar{T}^{-1}$) eigenvectors as well as the diagonal matrix of the eigenvalues $\bar{\Lambda}_c$ can be found in the Appendix A.9. The eigenvalues must be limited at stagnation points and sonic lines to prevent the dissipation from becoming zero. An efficient way of computing the product of $|\bar{A}_c|$ with $\vec{W}$ is provided in [35]. It should be noted that by setting $\epsilon^{(2)} = 1/2$ and $\epsilon^{(4)} = 0$, we obtain a first-order accurate, fully upwind scheme.

As it was already stated before, the idea here was to develop a scheme which accuracy is close to that of upwind schemes, but which is still computationally only slightly more expensive (about 15-20%) than the scalar dissipation approach. Results of comparisons with flux-vector splitting schemes (CUSP and AUSM) were recently reported in [37].

4.3.2 Flux-Vector Splitting Schemes

The flux-vector splitting methods can be viewed as the first level of upwind schemes, since they only account for the direction of wave propagation. The flux-vector splitting schemes decompose the vector of the convective fluxes into two parts – either according to the sign of certain characteristic variables, or into a convective and a pressure part. The well-known Van Leer's flux-vector splitting scheme [39] belongs to the first category based on characteristic decomposition. The second approach is followed by more recent methods like the Advection Upstream Splitting Method (AUSM) of Liou et al. [40], [41], or the Convective Upwind Split Pressure (CUSP) scheme of Jameson [42], [43], respectively. Further similar approaches are the Low-Diffusion Flux-Splitting Scheme (LDFSS) introduced by Edwards [44], or the Mach number-based Advection Pressure Splitting (MAPS) scheme of Rossow [45], [46].

The flux-vector splitting schemes can be implemented only in the framework of the cell-centred scheme (Subsection 4.2.1), or the cell-vertex scheme with dual control volumes (Subsection 4.2.3). Their advantage can be seen in only a moderately increased numerical effort but a much better resolution of shocks and boundary layers, as compared to the central scheme with scalar artificial dissipation. However, the matrix dissipation scheme (Eq. (4.58)), can also produce results of comparable accuracy [37]. Because of certain numerical difficulties, many modifications to the basic schemes (particularly to AUSM) were devised by various researchers and the development still continues. In the following, we shall present the basics of the Van Leer, AUSM and CUSP schemes, give some hints with respect to the most important modifications, and provide references to the corresponding literature.

Van Leer's Scheme

Van Leer's flux-vector splitting scheme [39] is based on characteristic decomposition of the convective fluxes. An extension of the approach to body-fitted grids was presented in [47], [48].

The convective flux is split into a positive and a negative part, i.e.,

$$\vec{F}_c = \vec{F}_c^+ + \vec{F}_c^- \,, \tag{4.60}$$

according to the Mach number normal to the face of the control volume (e.g., at $(I+1/2)$ - see Fig. 4.8)

$$(M_n)_{I+1/2} = \left(\frac{V}{c}\right)_{I+1/2} \,, \tag{4.61}$$

where V represents the contravariant velocity (2.22) and c the speed of sound, respectively. In the case of the cell-vertex scheme with dual control volumes, the cell indices have to be changed to node indices.

The values of the flow variables ρ, u, v, w, and p, respectively, have to be interpolated first to the faces of the control volume correspondingly to Eq. (4.19) or Eq. (4.39). Then, the positive fluxes are computed with the left state and the negative fluxes with the right state. The advection Mach number $(M_n)_{I+1/2}$ is obtained from the relation [39]

$$(M_n)_{I+1/2} = M_L^+ + M_R^- \,, \tag{4.62}$$

where the split Mach numbers are defined as

$$M_L^+ = \begin{cases} M_L & \text{if } M_L \geq +1 \\ \frac{1}{4}(M_L+1)^2 & \text{if } |M_L| < 1 \\ 0 & \text{if } M_L \leq -1 \end{cases} \,, \tag{4.63}$$

and

$$M_R^- = \begin{cases} 0 & \text{if } M_R \geq +1 \\ \frac{1}{4}(M_R - 1)^2 & \text{if } |M_R| < 1 \\ M_R & \text{if } M_R \leq -1 \end{cases} . \tag{4.64}$$

The Mach numbers M_L and M_R are evaluated using the left and right state, respectively, i.e.,

$$M_L = \frac{V_L}{c_L}, \quad M_R = \frac{V_R}{c_R} . \tag{4.65}$$

In the case of $|M_n| < 1$ (subsonic flow), the positive and the negative flux parts are given by

$$\vec{F}_c^{\pm} = \begin{bmatrix} f_{\text{mass}}^{\pm} \\ f_{\text{mass}}^{\pm}\left[n_x(-V \pm 2c)/\gamma + u\right] \\ f_{\text{mass}}^{\pm}\left[n_y(-V \pm 2c)/\gamma + v\right] \\ f_{\text{mass}}^{\pm}\left[n_z(-V \pm 2c)/\gamma + w\right] \\ f_{\text{energy}}^{\pm} \end{bmatrix} . \tag{4.66}$$

The mass and energy flux components are defined as

$$\begin{aligned} f_{\text{mass}}^{+} &= +\rho_L c_L \frac{(M_L + 1)^2}{4} \\ f_{\text{mass}}^{-} &= -\rho_R c_R \frac{(M_R - 1)^2}{4} \\ f_{\text{energy}}^{\pm} &= f_{\text{mass}}^{\pm} \left\{ \frac{[(\gamma - 1)V \pm 2c]^2}{2(\gamma^2 - 1)} + \frac{u^2 + v^2 + w^2 - V^2}{2} \right\}_{L/R} . \end{aligned} \tag{4.67}$$

For supersonic flow, i.e., for $|M_n| \geq 1$, the fluxes are evaluated from

$$\begin{aligned} \vec{F}_c^{+} &= \vec{F}_c \quad \vec{F}_c^{-} = 0 \quad &\text{if } M_n \geq +1 \\ \vec{F}_c^{+} &= 0 \quad \vec{F}_c^{-} = \vec{F}_c \quad &\text{if } M_n \leq -1 . \end{aligned} \tag{4.68}$$

The evaluation of the left and right state follows generally the MUSCL approach [29], which is given by Eqs. (4.46). The higher order schemes $\hat{\kappa} = -1$, $\hat{\kappa} = 0$ and $\hat{\kappa} = 1/3$, respectively, require a limiter if the flow field contains discontinuities like shocks. More details are provided in Subsection 4.3.5.

The flux-vector splitting scheme of Van Leer performs very well in the case of Euler equations. But several investigations [49], [50], carried out with the Navier-Stokes equations revealed that splitting errors in the momentum and the energy equations smear out the boundary layers and also lead to inaccurate stagnation and wall temperatures. A modification to the momentum flux in the direction normal to the boundary layer was therefore suggested in Ref. [51]. A similar remedy for the energy flux was proposed in Ref. [52]. Both modifications together remove the splitting errors, and hence they improve the solution accuracy considerably [53].

AUSM

The Advection Upstream Splitting Method (AUSM) was introduced by Liou and Steffen [40], and Liou [54]. It was subsequently modified by Wada and Liou [55] and renamed as AUSMD/V. Finally, an improved version termed AUSM$^+$ was presented by Liou [41], [56].

The underlying idea of the approach is based on the observation that the vector of convective fluxes (2.21) consists of two physically distinct parts, namely the convective and the pressure part

$$\vec{F}_c = V \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ \rho H \end{bmatrix} + \begin{bmatrix} 0 \\ n_x p \\ n_y p \\ n_z p \\ 0 \end{bmatrix} . \tag{4.69}$$

The first term in Eq. (4.69) represents scalar quantities, which are convected by the contravariant velocity V. By contrast, the pressure term is governed by the acoustic wave speed. The idea now is to discretise the convective term in purely upwind manner by taking either the left or the right state, depending on the sign of V (even for subsonic flow). On the other hand, the pressure term includes both states in the subsonic case. It becomes fully upwind for supersonic flow only.

Following the basic AUSM from [40], we introduce an advection Mach number $(M_n)_{I+1/2}$ from Eq. (4.61). Herewith, we can recast the convective flux at the face $(I+1/2)$, or $(i+1/2)$ of the control volume, respectively, into

$$(\vec{F}_c)_{I+1/2} = (M_n)_{I+1/2} \begin{bmatrix} \rho c \\ \rho c u \\ \rho c v \\ \rho c w \\ \rho c H \end{bmatrix}_{L/R} + \begin{bmatrix} 0 \\ n_x p \\ n_y p \\ n_z p \\ 0 \end{bmatrix}_{I+1/2} , \tag{4.70}$$

where

$$(\bullet)_{L/R} = \begin{cases} (\bullet)_L & \text{if } M_{I+1/2} \geq 0 \\ (\bullet)_R & \text{otherwise} \end{cases} . \tag{4.71}$$

Similar to Van Leer's flux-vector splitting scheme, the advection Mach number is evaluated as a sum of the left and right split Mach numbers according to the relations (4.62) and (4.63)-(4.65). The computation of the left and right state (flow quantities: ρ, u, v, w, p, H) is based again on a separate interpolation to the faces of the control volume accordingly to Eq. (4.19) or Eq. (4.39). The interpolation follows the MUSCL methodology [29], as it is given in Eq. (4.46). All higher-order MUSCL schemes ($\hat{\kappa} = -1$, $\hat{\kappa} = 0$, and $\hat{\kappa} = 1/3$) require a limiter, if the flow field contains strong gradients like shocks. Please refer to Subsection 4.3.5 for more details.

The pressure at the face $(I+1/2)$ of the control volume is obtained from the splitting [40]

$$p_{I+1/2} = p_L^+ + p_R^- \tag{4.72}$$

with the split pressures given by [39]

$$p_L^+ = \begin{cases} p_L & \text{if } M_L \geq +1 \\ \dfrac{p_L}{4}(M_L+1)^2(2-M_L) & \text{if } |M_L| < 1 \\ 0 & \text{if } M_L \leq -1 \end{cases}, \tag{4.73}$$

and

$$p_R^- = \begin{cases} 0 & \text{if } M_R \geq +1 \\ \dfrac{p_R}{4}(M_R-1)^2(2+M_R) & \text{if } |M_R| < 1 \\ p_R & \text{if } M_R \leq -1 \end{cases}. \tag{4.74}$$

It is also possible to use the following lower-order expansion for $|M_{L/R}| < 1$

$$p_{L/R}^{\pm} = \frac{p_{L/R}}{2}(1 \pm M_{L/R}). \tag{4.75}$$

It should be noted that we can write AUSM also in the form

$$(\vec{F}_c)_{I+1/2} = \frac{1}{2}(M_n)_{I+1/2}\left\{\begin{bmatrix}\rho c\\ \rho cu\\ \rho cv\\ \rho cw\\ \rho cH\end{bmatrix}_L + \begin{bmatrix}\rho c\\ \rho cu\\ \rho cv\\ \rho cw\\ \rho cH\end{bmatrix}_R\right\}$$

$$-\frac{1}{2}\left|(M_n)_{I+1/2}\right|\left\{\begin{bmatrix}\rho c\\ \rho cu\\ \rho cv\\ \rho cw\\ \rho cH\end{bmatrix}_R - \begin{bmatrix}\rho c\\ \rho cu\\ \rho cv\\ \rho cw\\ \rho cH\end{bmatrix}_L\right\} \tag{4.76}$$

$$+\begin{bmatrix}0\\ n_x(p_L^+ + p_R^-)\\ n_y(p_L^+ + p_R^-)\\ n_z(p_L^+ + p_R^-)\\ 0\end{bmatrix}.$$

The first term on the right-hand side of the above Eq. (4.76) represents a Mach number-weighted average of the left and right state – similar to the average of fluxes Eq. (4.15), (4.20) or Eq. (4.35), (4.40), respectively. The second term has a dissipative character. It is scaled by the scalar value $|(M_n)_{I+1/2}|$.

AUSM proved to deliver a crisp resolution of strong shocks and accurate results for boundary layers. However, the original AUSM [40], [54] was found to generate local pressure oscillations at shocks and in cases where the flow is aligned with the grid [57]. In [57], [58] it was therefore suggested to switch at shocks to Van Leer's scheme. When the advection Mach number $(M_n)_{I+1/2}$ tends to zero, the dissipation term in Eq. (4.76) will approach zero as well. Thus, any disturbances cannot be damped by the scheme. In order to solve the flow alignment problem, it was proposed in [57] to modify the scaling of the dissipation term as follows

$$|(M_n)_{I+1/2}| = \begin{cases} |(M_n)_{I+1/2}| & \text{if } |(M_n)_{I+1/2}| > \delta \\ \dfrac{(M_n)^2_{I+1/2} + \delta^2}{2\delta} & \text{if } |(M_n)_{I+1/2}| \leq \delta \end{cases}, \tag{4.77}$$

where δ is a small value ($0 < \delta \leq 0.5$). Hence, there will be always a sufficient amount of numerical dissipation. In order to retain the accuracy of AUSM for boundary layers, the parameter δ could be reduced in the wall normal direction using the same idea as given for the central scheme by Eq. (4.57).

Further improvements of the basic AUSM, with respect to better behaviour in the vicinity of shocks, was presented in [41], [56] as AUSM$^+$. The modifications consist of new Mach and pressure splittings, which replace the relations (4.63), (4.64) and (4.73), (4.74), respectively.

CUSP Scheme

The concept of the Convective Upwind Split Pressure (CUSP) scheme is quite similar to that of AUSM. But the CUSP approach has the advantage to be formulated as an average of fluxes (but without weighting like within AUSM) minus a dissipation term. This feature is crucial for the implementation in an explicit, hybrid multistage scheme. Furthermore, because of the different scaling factors compared to AUSM, the CUSP scheme behaves more favourably in the case of flow alignment.

The CUSP scheme was introduced by Jameson [42], [59], [60], and subsequently modified by Tatsumi et al. [43], [61]. It can be implemented either within the cell-centred or the (cell-vertex) dual control-volume type of spatial discretisation.

The convective fluxes (Eq. (2.21)) through a face of the control volume are approximated using arithmetic average of fluxes, according to the Equations (4.15), (4.20), (4.35), or (4.40), respectively. The dissipation term is then subtracted from the central fluxes for stabilisation. Thus, the total convective fluxes at the face (I+1/2) read

$$(\vec{F}_c)_{I+1/2} = \frac{1}{2}\left[\vec{F}_c(\vec{W}_R) + \vec{F}_c(\vec{W}_L)\right] - \vec{D}_{I+1/2}\,. \tag{4.78}$$

In the case of the dual control-volume discretisation, (i+1/2) would apply instead. The dissipation term, which is composed of a linear combination of the

differences of the state and the flux vector, can be expressed as

$$\vec{D}_{I+1/2} = \frac{1}{2}(\alpha^* c)_{I+1/2} \left\{ \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ \rho H \end{bmatrix}_R - \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ \rho H \end{bmatrix}_L \right\}$$
$$+ \frac{1}{2}\beta_{I+1/2} \left\{ \begin{bmatrix} \rho V \\ \rho u V + n_x p \\ \rho v V + n_y p \\ \rho w V + n_z p \\ \rho H V \end{bmatrix}_R - \begin{bmatrix} \rho V \\ \rho u V + n_x p \\ \rho v V + n_y p \\ \rho w V + n_z p \\ \rho H V \end{bmatrix}_L \right\}. \tag{4.79}$$

This particular formulation is denoted the H-CUSP scheme, since it preserves the total enthalpy [43]. The left (L) and right (R) state is evaluated similarly to the MUSCL approach [29], using the limited interpolation (4.114)-(4.117). The two parameters α^* and β are defined as

$$\alpha^* c = \begin{cases} |\tilde{V}| & \text{if } \beta = 0 \\ -(1+\beta)\Lambda^- & \text{if } \beta > 0 \text{ and } 0 < M_n < 1 \\ +(1-\beta)\Lambda^+ & \text{if } \beta < 0 \text{ and } -1 < M_n < 0 \\ 0 & \text{if } |M_n| \geq 1 \end{cases} \tag{4.80}$$

and

$$\beta = \begin{cases} +\max\left(0, \dfrac{\tilde{V} + \Lambda^-}{\tilde{V} - \Lambda^-}\right) & \text{if } 0 \leq M_n < 1 \\ -\max\left(0, \dfrac{\tilde{V} + \Lambda^+}{\tilde{V} - \Lambda^+}\right) & \text{if } -1 \leq M_n < 0 \\ \text{sign}(M_n) & \text{if } |M_n| \geq 1 \end{cases} \tag{4.81}$$

with $M_n = \tilde{V}/\tilde{c}$, correspondingly to Eq. (4.61). The positive and negative eigenvalues Λ^+ and Λ^- in Eqs. (4.80), (4.81) are given by [60]

$$\Lambda^{\pm} = \frac{\gamma+1}{2\gamma}\tilde{V} \pm \sqrt{\left(\frac{\gamma-1}{2\gamma}\tilde{V}\right)^2 + \frac{\tilde{c}^2}{\gamma}}. \tag{4.82}$$

where $\tilde{V}$ denotes the contravariant velocity at the face of the control volume (4.83), γ is the ratio of specific heat coefficients and $\tilde{c}$ stands for the speed of sound.

All flow variables in the above formulae (4.80)-(4.82) are obtained at the faces of the control volume using Roe averages [64], i.e.,

$$
\begin{aligned}
\tilde{u}_{I+1/2} &= \frac{u_L\sqrt{\rho_L} + u_R\sqrt{\rho_R}}{\sqrt{\rho_L} + \sqrt{\rho_R}} \\
\tilde{v}_{I+1/2} &= \frac{v_L\sqrt{\rho_L} + v_R\sqrt{\rho_R}}{\sqrt{\rho_L} + \sqrt{\rho_R}} \\
\tilde{w}_{I+1/2} &= \frac{w_L\sqrt{\rho_L} + w_R\sqrt{\rho_R}}{\sqrt{\rho_L} + \sqrt{\rho_R}} \\
\tilde{H}_{I+1/2} &= \frac{H_L\sqrt{\rho_L} + H_R\sqrt{\rho_R}}{\sqrt{\rho_L} + \sqrt{\rho_R}} \\
\tilde{c}_{I+1/2} &= \sqrt{(\gamma-1)\left(\tilde{H} - \frac{\tilde{u}^2 + \tilde{v}^2 + \tilde{w}^2}{2}\right)_{I+1/2}} \\
\tilde{V}_{I+1/2} &= \tilde{u}_{I+1/2}\, n_x + \tilde{v}_{I+1/2}\, n_y + \tilde{w}_{I+1/2}\, n_z\,.
\end{aligned}
\tag{4.83}
$$

The parameters α^* and β are defined such that full upwinding of the convective fluxes results for supersonic flow, i.e., $\alpha^* c = 0$ and $\beta = \text{sign}(M_n)$. On the other hand, in subsonic flow (when $\beta = 0$) the dissipation is scaled by $|\tilde{V}|$. This is a desirable property for the computation of viscous layers. In cases of large aspect ratio cells, explicit time-stepping schemes usually require increased numerical dissipation in the direction of the longer cell side in order to stay robust. This can be accomplished by employing ratios of spectral radii similar to Eq. (4.57). More details can be found in Ref. [37], which also contains comparisons between the CUSP scheme and the scalar as well as the matrix artificial dissipation (Subsection 4.3.1) schemes.

4.3.3 Flux-Difference Splitting Schemes

The flux-difference splitting schemes evaluate the convective fluxes at a face of the control volume from the (in general discontinuous) left and right state by solving the Riemann (shock tube) problem. The idea was first introduced by Godunov [62]. In contrast to the flux-vector splitting schemes, the flux-difference splitting considers not only the direction of wave (information) propagation, but also the waves themselves. In order to reduce the computational effort of Godunov's scheme for the exact solution of the Riemann problem, approximate Riemann solvers were developed, e.g., by Osher et al. [63] and Roe [64]. In particular, Roe's method is applied quite often because of its high accuracy for boundary layers and good resolution of shocks. Therefore, we shall present the Roe solver in more detail in the following subsection.

Roe Scheme

Roe's approximate Riemann solver can be implemented either in the framework of the cell-centred scheme or the dual control-volume scheme. It is based on the decomposition of the flux difference over a face of the control volume into a sum of wave contributions, while ensuring the conservation properties of the Euler equations. On the face $(I+1/2)$ or $(i+1/2)$, respectively, the difference is expressed as [64]

$$(\vec{F}_c)_R - (\vec{F}_c)_L = (\bar{A}_{Roe})_{I+1/2}\,(\vec{W}_R - \vec{W}_L)\,. \tag{4.84}$$

In the above Eq. (4.84), $\bar{A}_{Roe}$ denotes the so-called *Roe matrix*, and L or R the left and right state (see Fig. 4.8), respectively. The Roe matrix is identical to the convective flux Jacobian $\bar{A}_c$ (see Appendix A.7), where the flow variables are replaced by the so-called *Roe-averaged* variables. These are computed from the left and the right state by the formulae [64], [65]

$$\begin{aligned}
\tilde{\rho} &= \sqrt{\rho_L \rho_R} \\
\tilde{u} &= \frac{u_L\sqrt{\rho_L} + u_R\sqrt{\rho_R}}{\sqrt{\rho_L} + \sqrt{\rho_R}} \\
\tilde{v} &= \frac{v_L\sqrt{\rho_L} + v_R\sqrt{\rho_R}}{\sqrt{\rho_L} + \sqrt{\rho_R}} \\
\tilde{w} &= \frac{w_L\sqrt{\rho_L} + w_R\sqrt{\rho_R}}{\sqrt{\rho_L} + \sqrt{\rho_R}} \\
\tilde{H} &= \frac{H_L\sqrt{\rho_L} + H_R\sqrt{\rho_R}}{\sqrt{\rho_L} + \sqrt{\rho_R}} \\
\tilde{c} &= \sqrt{(\gamma - 1)\left(\tilde{H} - \tilde{q}^2/2\right)} \\
\tilde{V} &= \tilde{u}n_x + \tilde{v}n_y + \tilde{w}n_z \\
\tilde{q}^2 &= \tilde{u}^2 + \tilde{v}^2 + \tilde{w}^2\,.
\end{aligned} \tag{4.85}$$

We can make the decomposition into waves in Roe's scheme clearer when we insert the diagonalisation of the Roe matrix, i.e., $\bar{A}_{Roe} = \bar{T}\bar{\Lambda}_c\bar{T}^{-1}$, into the Eq. (4.84)

$$(\vec{F}_c)_R - (\vec{F}_c)_L = \bar{T}\bar{\Lambda}_c\,(\vec{C}_R - \vec{C}_L)\,. \tag{4.86}$$

The matrix of left $(\bar{T}^{-1})$ and right $(\bar{T})$ eigenvectors, as well as the diagonal matrix of eigenvalues $(\bar{\Lambda}_c)$ are evaluated using Roe's averaging. In the above Eq. (4.86), the characteristic variables $\vec{C}$ represent the wave amplitudes, the eigenvalues Λ_c are the associated wave speeds of the approximate Riemann problem, and finally the right eigenvectors are the waves themselves.

Following from the previous discussion, the convective fluxes are evaluated at the faces of a control volume faces corresponding to [64]

$$(\vec{F}_c)_{I+1/2} = \frac{1}{2}\left[\vec{F}_c(\vec{W}_R) + \vec{F}_c(\vec{W}_L) - |\bar{A}_{Roe}|_{I+1/2}(\vec{W}_R - \vec{W}_L)\right]. \tag{4.87}$$

The product of $|\bar{A}_{Roe}|$ and the difference of the left and right state can be evaluated as follows

$$|\bar{A}_{Roe}|(\vec{W}_R - \vec{W}_L) = |\Delta\vec{F}_1| + |\Delta\vec{F}_{2,3,4}| + |\Delta\vec{F}_5|, \tag{4.88}$$

where

$$|\Delta\vec{F}_1| = |\tilde{V} - \tilde{c}|\left(\frac{\Delta p - \tilde{\rho}\tilde{c}\Delta V}{2\tilde{c}^2}\right)\begin{bmatrix} 1 \\ \tilde{u} - \tilde{c}n_x \\ \tilde{v} - \tilde{c}n_y \\ \tilde{w} - \tilde{c}n_z \\ \tilde{H} - \tilde{c}\tilde{V} \end{bmatrix} \tag{4.89}$$

$$\begin{aligned} |\Delta\vec{F}_{2,3,4}| = |\tilde{V}| \Bigg\{ &\left(\Delta\rho - \frac{\Delta p}{\tilde{c}^2}\right)\begin{bmatrix} 1 \\ \tilde{u} \\ \tilde{v} \\ \tilde{w} \\ \tilde{q}^2/2 \end{bmatrix} \\ &+ \tilde{\rho}\begin{bmatrix} 0 \\ \Delta u - \Delta V n_x \\ \Delta v - \Delta V n_y \\ \Delta w - \Delta V n_z \\ \tilde{u}\Delta u + \tilde{v}\Delta v + \tilde{w}\Delta w - \tilde{V}\Delta V \end{bmatrix}\Bigg\} \end{aligned} \tag{4.90}$$

$$|\Delta\vec{F}_5| = |\tilde{V} + \tilde{c}|\left(\frac{\Delta p + \tilde{\rho}\tilde{c}\Delta V}{2\tilde{c}^2}\right)\begin{bmatrix} 1 \\ \tilde{u} + \tilde{c}n_x \\ \tilde{v} + \tilde{c}n_y \\ \tilde{w} + \tilde{c}n_z \\ \tilde{H} + \tilde{c}\tilde{V} \end{bmatrix}. \tag{4.91}$$

The jump condition is defined as $\Delta(\bullet) = (\bullet)_R - (\bullet)_L$ and the Roe-averaged variables are given in Eq. (4.85), respectively.

The left and the right state are determined using the MUSCL scheme [29], which is given in Eq. (4.46). All higher-order schemes ($\hat{\kappa} = -1$, $\hat{\kappa} = 0$, and $\hat{\kappa} = 1/3$) have to be supplemented by limiters (Subsection 4.3.5), if the flow field contains any discontinuities.

Because of the formulation in Eq. (4.84), Roe's approximate Riemann solver will produce an unphysical expansion shock in the case of stationary expansion, for which $(\vec{F}_c)_L = (\vec{F}_c)_R$ but $\vec{W}_L \neq \vec{W}_R$. Furthermore, the so-called "carbuncle phenomenon" may occur, where a perturbation grows ahead of a strong bow shock along the stagnation line [66], [67]. See also the discussion in Ref. [68]. The underlying difficulty is that the original scheme does not recognise the

sonic point. In order to solve this problem, the modulus of the eigenvalues $|\Lambda_c| = |\tilde{V} \pm \tilde{c}|$ can be modified using Harten's *entropy correction* [69], [70]

$$|\Lambda_c| = \begin{cases} |\Lambda_c| & \text{if } |\Lambda_c| > \delta \\ \dfrac{\Lambda_c^2 + \delta^2}{2\delta} & \text{if } |\Lambda_c| \leq \delta\,, \end{cases} \tag{4.92}$$

where δ is a small value, which can be conveniently set equal to some fraction (e.g., 1/10) of the local speed of sound. In order to prevent the linear waves $|\Delta \vec{F}_{2,3,4}|$ from disappearing for $\tilde{V} \to 0$ (e.g., at stagnation points or for grid-aligned flow), the above modification can also be applied to $|\tilde{V}|$.

A clear disadvantage of the Roe solver as compared to the central scheme or to the flux-vector splitting schemes shows up for a real gas simulation. Namely, the Roe matrix and averaging have to be changed correspondingly, which may become quite complicated. The reader may find examples of formulations for equilibrium as well as non-equilibrium real gas flows in [71]-[74] and in the references cited therein.

4.3.4 Total Variation Diminishing Schemes

The idea of Total Variation Diminishing (TVD) schemes was first pursued by Harten [75]. The TVD schemes are based on a concept aimed at preventing the generation of new extrema in the flow solution. The principal condition for a TVD scheme is that the total variation of the solution, defined as

$$\text{TV} \equiv \sum_I |U_{I+1} - U_I| \tag{4.93}$$

for a scalar conservation equation, decreases in time. This implies that maxima in the solution must be non-increasing and minima non-decreasing. Hence no new local extrema may be created during the time evolution. Thus, a discretisation methodology with TVD properties allows it to resolve strong shock waves accurately, without any spurious oscillations of the solution, as they are for example generated by the central scheme with scalar or matrix artificial dissipation (Subsection 4.3.1).

The TVD schemes are implemented as an average of the convective fluxes combined with an additional dissipation term (flux-limited dissipation), which complies with the TVD conditions [75], [76]. If the dissipation term depends on the sign of the characteristic speeds, we speak of a *symmetric* TVD scheme [77], [78], otherwise of an *upwind* TVD scheme [79]-[83]. Experience shows that the upwind TVD scheme offers higher accuracy than the symmetric TVD scheme [84]. The upwind TVD scheme is particularly suitable for the simulation of supersonic and hypersonic flow fields [85]. It is also capable of accurate resolution of boundary layers [53], especially if the modification described in Ref. [86] is applied.

Upwind TVD Scheme

In this framework, the convective fluxes through the face $(I+1/2)$ of the control volume (see Fig. 4.8) can be expressed as

$$(\vec{F}_c)_{I+1/2} = \frac{1}{2}\left[(\vec{F}_c)_{I+1} + (\vec{F}_c)_I\right] + \frac{1}{2}\bar{T}_{I+1/2}\vec{\Theta}_{I+1/2}\,. \tag{4.94}$$

In the case of the cell-vertex scheme with dual control volumes (Subsection 4.2.3), the indices would read $(i+1/2)$, $(i+1)$, etc. The matrix $\bar{T}$ contains the right eigenvectors of the Jacobian $\bar{A}_c = \partial\vec{F}_c/\partial\vec{W}$. The entries of the matrix can be found in the Appendix A.9. In Equation (4.94), the term $\vec{\Theta}$ takes account of the direction of the characteristic speeds. It controls the upwind direction of the difference operator. The l-th component of the vector $\vec{\Theta}$ is defined as (cf. [82])

$$\Theta^l_{I+1/2} = \frac{1}{2}\psi(\Lambda^l_{I+1/2})(\Psi^l_{I+1} + \Psi^l_I) - \psi(\Lambda^l_{I+1/2} + \chi^l_{I+1/2})\Delta C^l_{I+1/2}\,, \tag{4.95}$$

where Λ^l represents the individual eigenvalues of the diagonal matrix $\bar{\Lambda}_c$ (see Appendix A.9), and Ψ the limiter function (Eq. (4.118)), respectively. Furthermore,

$$\chi^l_{I+1/2} = \frac{1}{2}\psi(\Lambda^l_{I+1/2})\cdot\begin{cases}\dfrac{(\Psi^l_{I+1} - \Psi^l_I)}{\Delta C^l_{I+1/2}} & \text{if } \Delta C^l_{I+1/2} \neq 0\\[2ex] 0 & \text{if } \Delta C^l_{I+1/2} = 0\end{cases}\,, \tag{4.96}$$

and finally ΔC^l are the elements of the difference of characteristic variables, i.e.,

$$\Delta\vec{C}_{I+1/2} = \bar{T}^{-1}_{I+1/2}(\vec{W}_{I+1} - \vec{W}_I) \tag{4.97}$$

with $\bar{T}^{-1}$ being the matrix of left eigenvectors. The so-called *entropy correction* of Harten [69], [70],

$$\psi(z) = \begin{cases}|z| & \text{if } |z| > \delta_1\\[2ex] \dfrac{z^2+\delta_1^2}{2\delta_1} & \text{if } |z| \leq \delta_1\end{cases}\,, \tag{4.98}$$

prevents the value $\psi(z)$ from vanishing for $|z| \to 0$. The parameter δ_1 is best formulated as function of the velocity components and the speed of sound [82]

$$(\delta_1)_{I+1/2} = \delta\left(|u_{I+1/2}| + |v_{I+1/2}| + |w_{I+1/2}| + c_{I+1/2}\right), \tag{4.99}$$

where $0.05 \leq \delta \leq 0.5$. Values of the primitive variables at the face $(I+1/2)$ are obtained either from Roe's (4.85) or from simple arithmetic averaging of the states at I and $(I+1)$. The limiter function Ψ, which prevents the generation of spurious solutions near strong gradients, will be presented in the next subsection.

It should be stressed that the above upwind TVD scheme does not employ the MUSCL approach to achieve higher order accuracy.

One can show that the upwind TVD method is precisely of first-order in space when the limiter function Ψ in Eqs. (4.95), (4.96) is set equal to zero [87], which happens at discontinuities. Otherwise, the upwind TVD scheme, as presented above, is second-order accurate in smooth flow regions.

4.3.5 Limiter Functions

Second- and higher-order upwind spatial discretisations require the use of so-called *limiters* or *limiter functions* in order to prevent the generation of oscillations and spurious solutions in regions of high gradients (e.g., at shocks). Hence, what we are looking for is at least a *monotonicity preserving* scheme. This means that maxima in the flow field must be non-increasing, minima non-decreasing, and no new local extrema may be created during the time evolution. Or in other words, if the initial data is monotone then the solution has to remain monotone. The rather stringent conditions for monotonicity preserving schemes (or the more rigorous ones for TVD schemes) are often given up in favour of the *Local Extremum Diminishing* (LED) conditions [60]. Here, a local extremum contained only **within** the stencil has to decrease.

However, due to Godunov's theorem there is no possibility for a higher-order linear scheme (such as the MUSCL approach) to be monotonicity preserving [88]. It is therefore necessary to employ non-linear limiter functions in order to construct a monotonicity preserving or TVD discretisation. This is demonstrated in Fig. 4.10, where the upwind TVD scheme of Eq. (4.94) was used with and without a limiter to compute 2-D transonic flow past the NACA 0012 airfoil. It can be clearly seen that without limiter, the solution exhibits large oscillations in the neighbourhood of the shocks on the upper and the lower side of the airfoil. On the other hand, away from the shocks, the limited and the unlimited solutions become nearly identical.

The purpose of a limiter is to reduce the slopes (i.e., $(U_{I+1} - U_I)/\Delta x$) used to interpolate a flow variable to the face of a control volume, in order to constrain the solution variations. At strong discontinuities, the limiter has to reduce slopes to zero to prevent the generation of a new extremum. This implies for the MUSCL approach as well as for the TVD schemes that the (monotone) first-order upwind scheme ($\epsilon = 0$ in Eq. (4.46)) is recovered in the immediate vicinity of high gradients. The last requirement to be imposed on a limiter is quite obvious – the original unlimited discretisation has to be obtained in smooth flow regions, in order to keep the amount of numerical dissipation as low as possible. The effect of a limiter on the interpolation of the left and right states is sketched in Fig. 4.11. The example shows the slope reduction at the local minimum at I and the change of the slope at the cells $(I+1)$, $(I+2)$ to achieve a monotone solution. It is important to realise that a difference between the left and right state at a face may (and generally will) still be present.

In the following, we shall describe four different limiter functions, which are well-established and proven in practice. We shall consider limiters for the

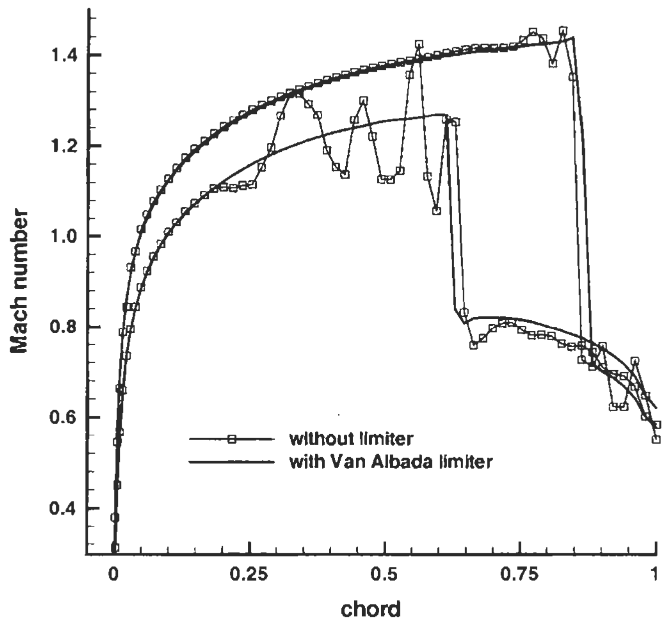


Figure 4.10: Comparison of inviscid transonic flow computation with and without limiter. NACA 0012 airfoil, $M_\infty = 0.85$, $\alpha = 1°$.

second-order MUSCL, for the CUSP and for the upwind TVD scheme.

Limiter Functions for MUSCL Interpolation

Van Leer's MUSCL approach [29] is turned into a monotonicity preserving scheme by employing a limiter function to reduce the differences $\Delta_+ U_I$ and $\Delta_- U_I$ in Eq. (4.47) when necessary. Herewith, the MUSCL interpolation formulae in Eq. (4.46) are modified as follows (see also Fig. 4.8)

$$
\begin{aligned}
U_R &= U_{I+1} - \frac{1}{4}\left[(1+\hat{\kappa})\Phi^+_{I+1/2}\Delta_- + (1-\hat{\kappa})\Phi^-_{I+3/2}\Delta_+\right] U_{I+1} \\
U_L &= U_I \quad + \frac{1}{4}\left[(1+\hat{\kappa})\Phi^-_{I+1/2}\Delta_+ + (1-\hat{\kappa})\Phi^+_{I-1/2}\Delta_-\right] U_I\,,
\end{aligned}
\tag{4.100}
$$

where the parameter ϵ was set equal to unity. The slope limiters are functions of ratios of consecutive solution variations, i.e., $\Phi^\pm_{I+1/2} = \Phi(r^\pm_{I+1/2})$, with [1]

$$
\begin{aligned}
r^+_{I+1/2} &= \frac{U_{I+2} - U_{I+1}}{U_{I+1} - U_I} \\
r^-_{I+1/2} &= \frac{U_I - U_{I-1}}{U_{I+1} - U_I}\,, \text{ etc.}
\end{aligned}
\tag{4.101}
$$

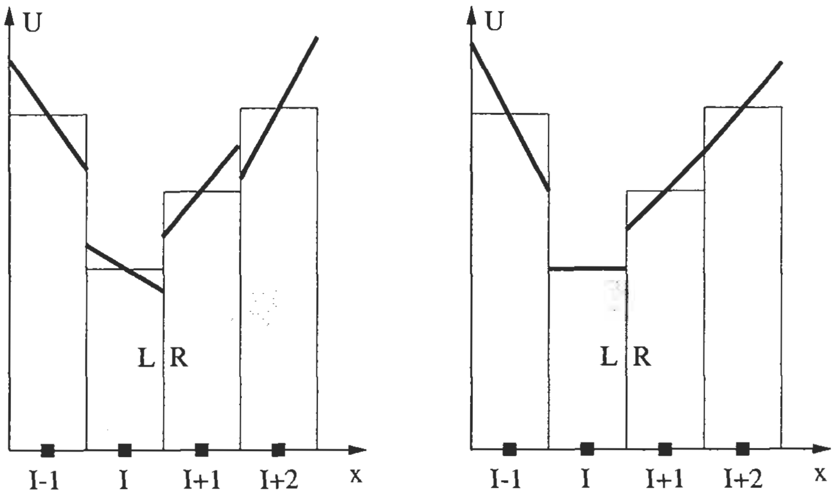


Figure 4.11: Comparison of direct (left) and limited (right) interpolation to the cell faces. Thick lines denote slopes $\Delta U/\Delta x$, bars represent values at cell centres.

If we substitute now r_L for $r^+_{I-1/2}$ and r_R for $r^-_{I+3/2}$, thus

$$\begin{aligned} r_R &= \frac{U_{I+1} - U_I}{U_{I+2} - U_{I+1}} = \frac{\Delta_-}{\Delta_+} U_{I+1} \\ r_L &= \frac{U_{I+1} - U_I}{U_I - U_{I-1}} = \frac{\Delta_+}{\Delta_-} U_I \, , \end{aligned} \tag{4.102}$$

we can write Eq. (4.100) in the form

$$\begin{aligned} U_R &= U_{I+1} - \frac{1}{4}\big[(1+\hat{\kappa}) r_R \, \Phi(1/r_R) + (1-\hat{\kappa})\Phi(r_R)\big]\,(U_{I+2} - U_{I+1}) \\ U_L &= U_I \quad + \frac{1}{4}\big[(1+\hat{\kappa}) r_L \, \Phi(1/r_L) + (1-\hat{\kappa})\Phi(r_L)\big]\,(U_I - U_{I-1}) \, . \end{aligned} \tag{4.103}$$

The above relationships Eq. (4.103) can be simplified if we consider only slope limiters with the symmetry property

$$\Phi(r) = \Phi(1/r) \, . \tag{4.104}$$

With this definition, the limited MUSCL interpolation Eq. (4.100) becomes [89]

$$\begin{aligned} U_R &= U_{I+1} - \frac{1}{2}\Psi_R \, (U_{I+2} - U_{I+1}) \\ U_L &= U_I \quad + \frac{1}{2}\Psi_L \, (U_I - U_{I-1}) \end{aligned} \tag{4.105}$$

with the **limiter function** defined as

$$\Psi_{L/R} = \frac{1}{2}\left[(1+\hat{\kappa})\, r_{L/R} + (1-\hat{\kappa})\right]\Phi_{L/R} \tag{4.106}$$

Different formulations of the slope limiter Φ in Eq. (4.106) are now possible, which can be tailored to specific values of $\hat{\kappa}$ to give the most accurate but stable and monotonicity preserving MUSCL scheme.

MUSCL scheme with $\hat{\kappa} = 0$

One particularly suitable combination for the second-order, upwind-biased scheme with $\hat{\kappa} = 0$ is [90]

$$\Phi(r) = \frac{2r}{r^2+1}\,. \tag{4.107}$$

In this case, the function $\Psi(r)$ corresponds to the Van Albada limiter [91]

$$\Psi(r) = \frac{r^2+r}{1+r^2}\,, \tag{4.108}$$

and we obtain with Eq. (4.105) the following expressions for the left and right state

$$\begin{aligned} U_R &= U_{I+1} - \frac{1}{2}\,\delta_R \\ U_L &= U_I \quad\ + \frac{1}{2}\,\delta_L \end{aligned} \tag{4.109}$$

The function δ is formally identical for both states. It reads

$$\delta = \frac{a(b^2+\epsilon) + b(a^2+\epsilon)}{a^2+b^2+2\epsilon}\,. \tag{4.110}$$

The coefficients a and b are defined for the left and right state as

$$\begin{aligned} a_R &= \Delta_+ U_{I+1}\,, \quad & b_R &= \Delta_- U_{I+1}\,, \\ a_L &= \Delta_+ U_I\,, & b_L &= \Delta_- U_I \end{aligned} \tag{4.111}$$

and the difference operators $\Delta_\pm$ are given by Eq. (4.47). The additional parameter ϵ in Eq. (4.110) prevents the activation of the limiter in smooth flow regions due to small-scale oscillations [90]. This is sometimes necessary in order to achieve a fully converged steady-state solution. The parameter ϵ is conveniently set proportional to the local grid scale, in 3D for example to $\Omega^{1/3}$ [90], [92]. Additional scaling of the parameter ϵ is required if the particular state variable U is given in physical units. It can be shown that the relations in Eq. (4.109) are identical to the original (unlimited) MUSCL scheme (4.46) with $\hat{\kappa} = 0$ in smooth regions. Thus, the accuracy of the solution is not influenced. On the other hand, the function δ becomes zero at local extrema, reducing the accuracy to first order as desired.

MUSCL scheme with $\hat{\kappa} = 1/3$

Another limiter function was devised for the three-point, second-order accurate upwind-biased MUSCL scheme with $\hat{\kappa} = 1/3$. Here, the slope limiter is given by

$$\Phi(r) = \frac{3r}{2r^2 - r + 2}\,. \tag{4.112}$$

In this case, the function $\Psi(r)$ corresponds to the limiter of Hemker and Koren [93]. Following the same way as in the previous case, we obtain formulae for the left and right state at the face $(I+1/2)$ which are identical to Eq. (4.109), but now with [90]

$$\delta = \frac{(2a^2 + \epsilon)b + (b^2 + 2\epsilon)a}{2a^2 + 2b^2 - ab + 3\epsilon}\,. \tag{4.113}$$

The definitions of the coefficients a, b, and of the parameter ϵ are retained.

Limiter for CUSP Scheme

In the framework of the CUSP scheme (Subsection 4.3.2), the left (L) and right (R) states are evaluated to second-order accuracy according to [43]

$$\begin{aligned} U_R &= U_{I+1} - \frac{1}{2} L\left(\Delta U_{I+3/2},\, \Delta U_{I-1/2}\right) \\ U_L &= U_I \quad + \frac{1}{2} L\left(\Delta U_{I+3/2},\, \Delta U_{I-1/2}\right)\,, \end{aligned} \tag{4.114}$$

where

$$\begin{aligned} \Delta U_{I+1/2} &= U_{I+1} - U_I \\ \Delta U_{I+3/2} &= U_{I+2} - U_{I+1}\,. \end{aligned} \tag{4.115}$$

In the above Eqs. (4.114) and (4.115), U represents a dependent variable and $L()$ the limited average

$$L(\Delta_1,\, \Delta_2) = \frac{1}{2}\,\Psi(\Delta_1,\, \Delta_2)\,(\Delta_1 + \Delta_2)\,, \tag{4.116}$$

respectively. The limiter itself is defined as

$$\Psi(\Delta_1,\, \Delta_2) = 1 - \left|\frac{\Delta_1 - \Delta_2}{|\Delta_1| + |\Delta_2| + \epsilon}\right|^{\sigma}\,, \tag{4.117}$$

where σ is a positive coefficient which is usually set equal to two. The constant ϵ is required to prevent division by zero (e.g., $\epsilon = 10^{-20}$). If Δ_1 and Δ_2 happen to have opposite sign but the same magnitude, the limiter becomes $\Psi = 0$. This means that we obtain only a first-order accurate approximation for the left and the right state.

Limiter for TVD Scheme

In comparison to the previous cases, the limiter here acts not on the conservative or the primitive variables, but on the characteristic variables $\vec{C}$. One particularly suitable limiter function is given by [82]

$$\Psi_I^l = \frac{\Delta C_{I-1/2}^l \Delta C_{I+1/2}^l + |\Delta C_{I-1/2}^l \Delta C_{I+1/2}^l|}{\Delta C_{I-1/2}^l + \Delta C_{I+1/2}^l + \epsilon}\,, \tag{4.118}$$

where the $\Delta C_{I+1/2}^l$ represents the difference of the characteristic variables at face $(I+1/2)$ of the control volume (Eq. (4.97)). The positive constant $\epsilon \approx 10^{-20}$ in the denominator prevents division by zero. In regions with high gradients, the limiter function becomes zero, which leads with Eq. (4.95) and (4.94) to first-order accurate upwind scheme. The upwind TVD scheme (4.94) retains second-order accuracy in areas of smooth flow, where $\Psi_I^l = C_I^l - C_{I-1}^l$.

4.4 Discretisation of Viscous Fluxes

The control volume for the viscous fluxes is generally chosen to be the same as for the convective fluxes in order to obtain a consistent spatial discretisation. An exception is made only in the case of the cell-vertex scheme with overlapping control volumes (Subsection 4.2.2), where the dual control volume (Subsection 4.2.3) is employed instead, primarily due to stability reasons [94]-[96]. The viscous fluxes $\vec{F}_v$ in the discretised governing equations (4.2) are, similar to Eqs. (4.17), (4.22), (4.37), (4.42), evaluated from variables averaged at the faces of the control volume. This is in line with the elliptic nature of the viscous fluxes. Thus, values of the velocity components (u, v, w), the dynamic viscosity μ, and of the heat conduction coefficient k, which are required for the computation of the viscous terms (2.23), (2.24) and of the stresses (2.15), are simply averaged at a face. In the case of the cell-centred scheme (Figs. 4.3 and 4.8), the values at the face $(I+1/2)$ of the control volume result from

$$U_{I+1/2} = \frac{1}{2}(U_I + U_{I+1}) \,, \tag{4.119}$$

where U is any of the above flow variables. The same holds in the case of both cell-vertex schemes for the face $(i+1/2)$ – see Figs. 4.5 and 4.8, respectively.

The remaining task is the evaluation of the first derivatives (gradients) of the velocity components in Eq. (2.15) and of temperature in Eq. (2.24). This can be accomplished in one of two ways, i.e., by using

- finite differences, or
- Green's theorem.

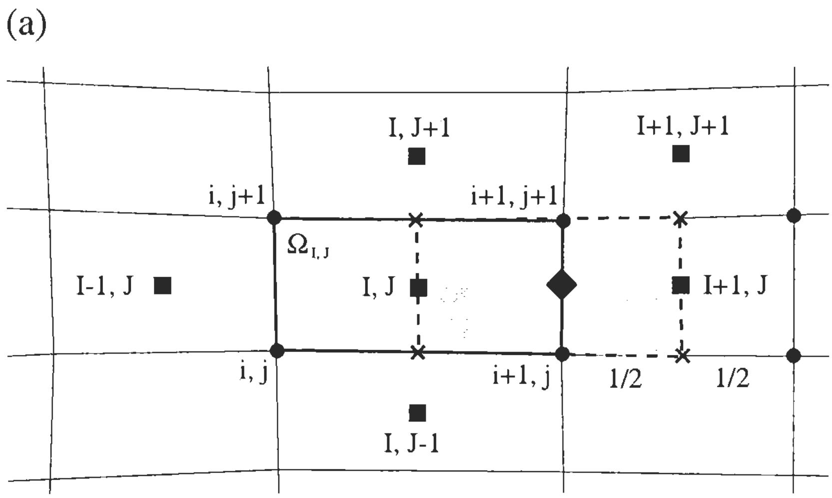
The first approach applies a local transformation from Cartesian coordinates (x, y, z) to the curvilinear coordinates (ξ, η, ζ), e.g.,

$$\frac{\partial U}{\partial x} = \frac{\partial U}{\partial \xi}\frac{\partial \xi}{\partial x} + \frac{\partial U}{\partial \eta}\frac{\partial \eta}{\partial x} + \frac{\partial U}{\partial \zeta}\frac{\partial \zeta}{\partial x} \,, \text{ etc.} \tag{4.120}$$

The derivatives U_ξ, U_η and U_ζ are obtained from finite difference approximations. More details can be found in Refs. [94]-[96]. See Appendix A.1 for the derivatives of the coordinates and for the Jacobian of the transformation.

Here, we prefer the second approach, which is more in line with the finite volume methodology treated in this book. However, it requires the construction of an additional control volume for the computation of the derivatives. This will be discussed below for the cell-centred and the cell-vertex scheme.

Once we obtained the values of the flow variables and of the first derivatives at the faces of the control volume, we can sum up the contributions due to the viscous fluxes according to Eq. (4.2). By adding the sum of the contributions to the inviscid fluxes, we completed the spatial discretisation, and we can thus integrate the approximated governing equations in time.



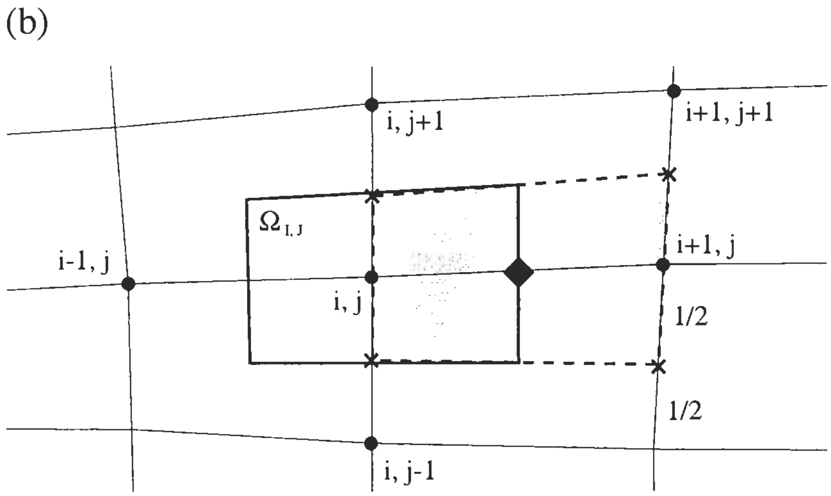


Figure 4.12: Auxiliary control volume Ω' (filled) for evaluation of first derivatives in two dimensions: (a) cell-centred scheme; (b) cell-vertex scheme. The diamond symbol denotes location where first derivatives are to be evaluated.

4.4.1 Cell-Centred Scheme

In order to apply Green's theorem, which relates the volume integral of the first derivative to the surface integral of U, we have to define a suitable control volume first. Since we need the derivatives at the midpoints of the faces for the summation in Eq. (4.2), we construct an auxiliary control volume centred at the face by connecting the midpoints of the edges defining adjacent grid cells [31], [19], [96] as shown in Fig. 4.12a. In order to evaluate the first derivative at the face $(I+1/2)$ – marked by a diamond symbol in Fig. 4.12a – we have to integrate the corresponding flow variable U over the boundary of the auxiliary control volume (denoted by the superscript $'$ in the following). Thus, e.g., for the derivative in the x-direction

$$\frac{\partial U}{\partial x} = \frac{1}{\Omega'} \int_{\partial\Omega'} U \, dS'_x \approx \frac{1}{\Omega'} \sum_{m=1}^{N_F} U_m \, S'_{x,m} \,, \tag{4.121}$$

where N_F stands for the number of faces ($N_F = 4$ in 2D and $N_F = 6$ in 3D). The volume Ω' and the components of the face vector $\vec{S}'_m = [S'_{x,m}, S'_{y,m}, S'_{z,m}]^T$, respectively, are computed as already presented in Section 4.1. The face values U_m are obtained either directly as cell-centred values (i.e., $U_{i,j}$ and $U_{i+1,j}$ on the left and the right face), or by averaging like on the upper and the lower face, e.g., at $J+1/2$

$$U_{m_{I,J+1/2}} = \frac{1}{4}\left(U_{I,J} + U_{I+1,J} + U_{I,J+1} + U_{I+1,J+1}\right) \,, \text{ etc.} \tag{4.122}$$

We can apply the same approach in three dimensions, where again four cell-centred values can be utilised for the averaging. Hence,

$$\begin{aligned} U_{m_{I,J+1/2,K}} &= \frac{1}{4}\left(U_{I,J,K} + U_{I+1,J,K} + U_{I,J+1,K} + U_{I+1,J+1,K}\right) \,, \\ U_{m_{I,J+1/2,K+1/2}} &= \frac{1}{4}\left(U_{I,J,K} + U_{I,J,K+1} + U_{I,J+1,K} + U_{I,J+1,K+1}\right) \,, \\ &\ldots \end{aligned} \tag{4.123}$$

The above scheme is quite compact, with the computational stencil extending over only nine cells in two dimensions and over 15 in three dimensions. It should be noted that this approach for computing the first derivatives cannot suppress the generation of two types of spurious modes (decoupled solutions at neighbouring cell centres) [97], [19]: the chequer-board mode, arising from the form of the integral around the control volume, and a pair of corrugated or washboard modes, arising from the averaging of values in neighbouring cells. However, there are generally no difficulties with this in practice. A more serious problem would occur, if the gradients would be first evaluated for each cell (similar to the convective fluxes) and then averaged at the cell faces. Although this approach may appear more attractive than the current methodology, it is not recommended since it leads to strong odd-even decoupling.

A disadvantage of above scheme is a loss of accuracy if the grid is not uniform [96], [97]. Namely, for arbitrarily stretched grids the approximation of the derivatives becomes inconsistent. Thus, the viscous fluxes are discretised with second-order accuracy only for smoothly stretched grids.

Finally, it should be noted that the TSL approximation of the Navier-Stokes equations (Subsection 2.4.3) can easily be realised by omitting the appropriate contributions when computing the gradients. For example, if the boundary layer would be oriented along the I-direction in Fig. 4.12a, contributions from the left (I, J) and the right side $(I+1, J)$ of the auxiliary control volume would be dropped.

4.4.2 Cell-Vertex Scheme

As already mentioned, both types of cell-vertex schemes resort to the dual control volume (Subsection 4.2.3) for the discretisation of the viscous fluxes. Hence, the question is how to evaluate the first derivatives at the faces of this control volume. Considering Fig. 4.12b, one possible alternative is to calculate the gradients at the cell centres first by integrating over the grid cells, which yields first-order accuracy on arbitrarily stretched grids. In a next step, the cell-based gradients are averaged at the faces of the control volume Ω [31], [98]. However, this approach cannot prevent an odd-even decoupling of the solution.

Another possibility, similar to the cell-centred scheme, is to construct an auxiliary control volume around the face by connecting the midpoints of the edges defining adjacent grid cells [99], [100]. This is depicted in Fig. 4.12b. The evaluation of the first differences proceeds along the same lines as discussed for the cell-centred scheme, with averaged quantities where necessary. It should be noted that this approach is formally identical to the finite difference approximation [94]-[96]. This scheme leads to first-order accurate discretisation of the viscous fluxes on arbitrarily stretched grids and to second-order accuracy on smooth grids [94], [96]. Another positive feature is that the computational stencil is confined to only nine nodes in two dimensions and to 15 nodes in three dimensions.

Finally, one further approach should be mentioned, where a more complex integration path was chosen, with averaging incorporating all neighbouring nodes [20]. A serious disadvantage of this scheme is that it encompasses a 25-point stencil even in two dimensions, which adds in general more numerical diffusion than more compact stencils. Furthermore, if an implicit scheme would be envisioned for the time integration, the bandwidth of the flux Jacobian would become prohibitively large. A detailed discussion of various methodologies for the gradient evaluation can also be found in Ref. [101].

Bibliography

[1] Hirsch, C.: *Numerical Computation of Internal and External Flows.* Vols. 1 and 2, John Wiley and Sons, 1988.

[2] Mohanraj, R.; Neumeier, Y.; Zinn, B.T.: *Characteristic-Based Treatment of Source Terms in Euler Equations for Roe Scheme.* AIAA Journal, 37 (1999), pp. 417-424.

[3] Jameson, A.; Baker, T.J.: *Solution of the Euler Equations for Complex Configurations.* AIAA Paper 83-1929, 1983.

[4] Rossow, C.-C.: *Berechnung von Strömungsfeldern durch Lösung der Euler-Gleichungen mit einer erweiterten Finite-Volumen Diskretisierungsmethode (Calculation of Flow Fields by the Solution of Euler Equations Using an Extended Finite Volume Discretisation Scheme).* DLR Research Report, No. 89-38, 1989.

[5] Bruner, C.W.S.: *Geometric Properties of Arbitrary Polyhedra in Terms of Face Geometry.* AIAA Journal, 33 (1995), p. 1350.

[6] Jameson, A.; Schmidt, W.; Turkel, E.: *Numerical Solutions of the Euler Equations by Finite Volume Methods Using Runge-Kutta Time-Stepping Schemes.* AIAA Paper 81-1259, 1981.

[7] Ni, R.H.: *Multiple Grid Scheme for Solving the Euler Equations.* AIAA Paper 81-1025, 1981.

[8] Hall, M.G.: *Cell-Vertex Multigrid Scheme for Solution of the Euler Equations.* Proc. Int. Conf. on Numerical Methods for Fluid Dynamics, Reading, UK, Springer Verlag, 1985.

[9] Radespiel, R.: *A Cell-Vertex Multigrid Method for the Navier-Stokes Equations.* NASA TM-101557, 1989.

[10] Radespiel, R.; Rossow, C.-C.; Swanson, R.C.: *An Efficient Cell-Vertex Multigrid Scheme for the Three-Dimensional Navier-Stokes Equations.* AIAA Paper 89-1953, 1989; also in AIAA Journal 28 (1990), pp. 1464-1472.

[11] Rossow, C.-C.: *Flux Balance Splitting - A New Approach for a Cell-Vertex Upwind Scheme.* Proc. 12th Int. Conf. on Numerical Methods in Fluid Dynamics, Oxford, UK, Springer Verlag, 1990.

[12] Rossow, C.-C.: *Accurate Solution of the 2D Euler Equations with an Efficient Cell-Vertex Upwind Scheme.* AIAA Paper 93-0071, 1993.

[13] Usab, W.J.: *Embedded Mesh Solutions of the Euler Equations Using a Multiple-Grid Method.* Ph.D. Thesis, MIT, Cambridge, Massachusetts, USA, 1983.

[14] Sidilkover, D.: *A Genuinely Multidimensional Upwind Scheme and Efficient Multigrid Solver for the Compressible Euler Equations.* ICASE Report, No. 94-84, 1994.

[15] Paillére, H.; Deconinck, H.; Roe, P.L.: *Conservative Upwind Residual-Distribution Schemes Based on the Steady Characteristics of the Euler Equations.* AIAA Paper 95-1700, 1995.

[16] Issman, E.; Degrez, G.; Deconinck, H.: *Implicit Upwind Residual-Distribution Euler and Navier-Stokes Solver on Unstructured Meshes.* AIAA Journal, 34 (1996), pp. 2021-2028.

[17] Van der Weide, E.; Deconinck, H.: *Compact Residual-Distribution Scheme Applied to Viscous Flow Simulations.* VKI Lecture Series 1998-03, 1998.

[18] Dick, E.: *A Flux-Difference Splitting Method for Steady Euler Equations.* J. Computational Physics, 76 (1988), pp. 19-32.

[19] Hall, M.: *A Vertex-Centroid Scheme for Improved Finite-Volume Solution of the Navier-Stokes Equations.* AIAA Paper 91-1540, 1991.

[20] Crumpton, P.I.; Shaw, G.J.: *A Vertex-Centred Finite Volume Method with Shock Detection.* Int. J. Numerical Methods in Fluids, 18 (1994), pp. 605-625.

[21] Roe, P.L.: *Error Estimates for Cell-Vertex Solutions of the Compressible Euler Equation.* ICASE Report No. 87-6, 1987.

[22] Hoffman, J.D.: *Relationship between the Truncation Errors of Centered Finite-Difference Approximations on Uniform and Nonuniform Meshes.* J. Computational Physics, 46 (1982), pp. 464-474.

[23] Arts, T.: *On the Consistency of four Different Control Surfaces Used for Finite Area Blade-to-Blade Calculations.* Int. J. Numerical Methods in Fluids, 4 (1984), pp. 1083-1096.

[24] Turkel, E.: *Accuracy of Schemes with Non-Uniform Meshes for Compressible Fluid Flows.* ICASE Report No. 85-43, 1985.

[25] Turkel, E.; Yaniv, S.; Landau, U.: *Accuracy of Schemes for the Euler Equations with Non-Uniform Meshes.* ICASE Report No. 85-59, 1985.

[26] Rossow, C.-C.: *Comparison of Cell-Centered and Cell-Vertex Finite Volume Schemes.* Proc. 7th GAMM Conference, Notes on Numerical Fluid Mechanics, Vieweg Publishing, 1987.

[27] Venkatakrishnan, V.: *Implicit Schemes and Parallel Computing in Unstructured Grid CFD.* ICASE Report No. 95-28, 1995.

[28] Venkatakrishnan, V.; Mavriplis, D.J.: *Implicit Method for the Computation of Unsteady Flows on Unstructured Grids.* J. Computational Physics, 127 (1996), pp. 380-397.

[29] Van Leer, B.: *Towards the Ultimate Conservative Difference Scheme V. A second Order Sequel to Godunov's method.* J. Computational Physics, 32 (1979), pp. 101-136.

[30] Leonard, B.P.: *Comparison of Truncation Error of Finite-Difference and Finite-Volume Formulations of Convection Terms.* NASA TM-105861, 1992.

[31] Martinelli, L.: *Calculation of Viscous Flows with a Multigrid Method.* Ph.D. Thesis, Dept. of Mech. and Aerospace Eng., Princeton University, 1987.

[32] Martinelli, L.; Jameson, A.: *Validation of a Multigrid Method for Reynolds Averaged Equations.* AIAA Paper 88-0414, 1988.

[33] Yoon, S.; Kwak, D.: *Artificial Dissipation Models for Hypersonic External Flow.* AIAA Paper 88-3708, 1988.

[34] Turkel, E.; Swanson, R.C.; Vatsa, V.N.; White, J.A.: *Multigrid for Hypersonic Viscous Two- and Three-Dimensional Flows.* AIAA Paper 91-1572, 1991.

[35] Swanson, R.C.; Turkel, E.: *On Central Difference and Upwind Schemes.* J. Computational Physics, 101 (1992), pp. 292-306.

[36] Jameson, A.: *Artificial Diffusion, Upwind Biasing, Limiters and their Effect on Accuracy and Multigrid Convergence in Transonic and Hypersonic Flow.* AIAA Paper 93-3559, 1993.

[37] Swanson, R.C.; Radespiel, R.; Turkel, E.: *Comparison of Several Dissipation Algorithms for Central Difference Schemes.* ICASE Report No. 97-40, 1997.

[38] Pulliam, T.H.: *Artificial Dissipation Models for the Euler Equations.* AIAA Journal, 24 (1986), pp. 1931-1940.

[39] Van Leer, B.: *Flux-Vector Splitting for the Euler Equations.* Proc. 8th Int. Conf. on Numerical Methods in Fluid Dynamics, Springer Verlag, 1982, pp. 507-512; also ICASE Report 82-30, 1982.

[40] Liou, M.-S.; Steffen, C.J. Jr.: *A New Flux Splitting Scheme.* NASA TM-104404, 1991; also J. Computational Physics, 107 (1993), pp. 23-39.

[41] Liou, M.-S.: *A Sequel to AUSM: $AUSM^+$.* J. Computational Physics, 129 (1996), pp. 364-382.

[42] Jameson, A.: *Artificial Diffusion, Upwind Biasing, Limiters and their Effect on Accuracy and Multigrid Convergence in Transonic and Hypersonic Flow.* AIAA Paper 93-3559, 1993.

[43] Tatsumi, S.; Martinelli, L.; Jameson, A.: *A New High Resolution Scheme for Compressible Viscous Flow with Shocks.* AIAA Paper 95-0466, 1995.

[44] Edwards, J.R.: *A Low-Diffusion Flux-Splitting Scheme for Navier-Stokes Calculations.* Computers & Fluids, 26 (1997), pp. 653-659.

[45] Rossow, C.-C.: *A Simple Flux Splitting Scheme for Compressible Flows.* Proc. 11th DGLR-Fach-Symposium, Berlin, Germany, November 10-12, 1998.

[46] Rossow, C.-C.: *A Flux Splitting Scheme for Compressible and Incompressible Flows.* AIAA Paper 99-3346, 1999.

[47] Thomas, J.L.; van Leer, B.; Walters, R.W.: *Implicit Flux-Split Schemes for the Euler Equations.* AIAA Paper 85-1680, 1985.

[48] Anderson, W.K.; Thomas, J.L.; van Leer, B.: *A Comparison of Finite Volume Flux Vector Splittings for the Euler Equations.* AIAA Journal, 24 (1986), pp. 1453-1460.

[49] Hänel, D.; Schwane, R.; Seider, G.: *On the Accuracy of Upwind Schemes for the Solution of the Navier-Stokes Equations.* AIAA Paper 87-1105, 1987.

[50] Van Leer, B.; Thomas, J.L.; Roe, P.L.; Newsome, R.W.: *A Comparison of Numerical Flux Formulas for the Euler and Navier-Stokes Equations.* AIAA Paper 87-1104, 1987.

[51] Hänel, D.; Schwane, R.: *An Implicit Flux-Vector Scheme for the Computation of Viscous Hypersonic Flows.* AIAA Paper 89-0274, 1989.

[52] Van Leer, B.: *Flux Vector Splitting for the 1990's.* Invited Lecture for the CFD Symposium on Aeropropulsion, Cleveland, Ohio, 1990.

[53] Seider, G.; Hänel, D.: *Numerical Influence of Upwind TVD Schemes on Transonic Airfoil Drag Prediction.* AIAA Paper 91-0184, 1991.

[54] Liou, M.S.: *On a New Class of Flux Splittings.* Proc. 13th Int. Conf. on Numerical Methods in Fluid Dynamics, Rome, Italy, 1992.

[55] Wada, Y.; Liou, M.-S.: *A Flux Splitting Scheme with High-Resolution and Robustness for Discontinuities.* AIAA Paper 94-0083, 1994.

[56] Liou, M.-S.: *Progress Towards an Improved CFD Method - AUSM$^+$.* AIAA Paper 95-1701, 1995.

[57] Radespiel, R.; Kroll, N.: *Accurate Flux Vector Splitting for Shocks and Shear Layers.* J. Computational Physics, 121 (1995), pp. 66-78.

[58] Edwards, J.R.: *Numerical Implementation of a Modified Liou-Steffen Upwind Scheme.* AIAA Journal, 32 (1994), pp. 2120-2122.

[59] Jameson, A.: *Positive Schemes and Shock Modelling for Compressible Flow.* Int. J. Numerical Methods in Fluids, 20 (1995), pp. 743-776.

[60] Jameson, A.: *Analysis and Design of Numerical Schemes for Gas Dynamics II: Artificial Diffusion and Discrete Shock Structure.* Int. J. Computational Fluid Dynamics, 5 (1995), pp. 1-38.

[61] Tatsumi, S.; Martinelli, L.; Jameson, A.: *A Design, Implementation and Validation of Flux Limited Schemes for the Solution of the Compressible Navier-Stokes Equations.* AIAA Paper 94-0647, 1994.

[62] Godunov, S.K.: *A Difference Scheme for Numerical Computation Discontinuous Solution of Hydrodynamic Equations.* Math. Sbornik (in Russian), 47 (1959), pp. 271-306; translated US Joint Publ. Res. Service, JPRS 7226, 1969.

[63] Osher, S.; Solomon, F.: *Upwind Difference Schemes for Hyperbolic Systems of Conservation Laws.* Math. Comp., 38 (1982), pp. 339-374.

[64] Roe, P.L.: *Approximate Riemann Solvers, Parameter Vectors, and Difference Schemes.* J. Computational Physics, 43 (1981), pp. 357-372.

[65] Roe, P.L.; Pike, J.: *Efficient Construction and Utilisation of Approximate Riemann Solutions.* Computing Methods in Applied Sciences and Engineering, R. Glowinski, J.L. Lions (eds.), North Holland Publishing, The Netherlands, 1984.

[66] Peery, K.M.; Imlay, S.T.: *Blunt-Body Flow Simulations.* AIAA Paper 88-2904, 1988.

[67] Lin, H.C.: *Dissipation Additions to Flux-Difference Splitting.* AIAA Paper 91-1544, 1991.

[68] Quirk, J.J.: *A Contribution to the Great Riemann Solver Debate.* ICASE Report No. 92-64, 1992.

[69] Harten, A.; Lax, P.D.; Van Leer, B.: *On Upstream Differencing and Godunov-Type Schemes for Hyperbolic Conservation Laws.* Soc. Industrial and Applied Mathematics Rew., 25 (1983), No. 1

[70] Harten, A.; Hyman, J.M.: *Self Adjusting Grid Methods for One-Dimensional Hyperbolic Conservation Laws.* J. Computational Physics, 50 (1983), pp. 235-269.

[71] Vinokur, M.: *Flux Jacobian Matrices and Generalized Roe Average for an Equilibrium Real Gas.* NASA CR-177512, 1988.

[72] Vinokur, M.; Montagné, J.-L.: *Generalized Flux-Vector Splitting and Roe Average for an Equilibrium Real Gas.* J. Computational Physics, 89 (1990), pp. 276-300.

[73] Grossman, B.; Cinnella, P.: *Flux-Split Algorithms for Flows with Non-equilibrium Chemistry and Vibrational Relaxation.* J. Computational Physics, 88 (1990), pp. 131-168.

[74] Shuen, J.-S.; Liou, M.-S.; Van Leer, B.: *Inviscid Flux-Splitting Algorithms for Real Gases with Non-equilibrium Chemistry.* J. Computational Physics, 90 (1990), pp. 371-395.

[75] Harten, A.: *High Resolution Schemes for Hyperbolic Conservation Laws.* J. Computational Physics, 49 (1983), pp. 357-393.

[76] Jameson, A.; Lax, P.D.: *Conditions for the Construction of Multi-Point Total Variation Diminishing Difference Schemes.* MAE Report 1650, Dept. of Mechanical and Aerospace Engineering, Princeton University, 1984.

[77] Davis, S.F.: *TVD Finite Difference Schemes and Artificial Viscosity.* ICASE Report No. 84-20, 1984.

[78] Yee, H.C.: *Construction of Implicit and Explicit Symmetric TVD Schemes and Their Applications.* J. Computational Physics, 68 (1987), pp. 151-179.

[79] Yee, H.C.; Kutler, P.: *Application of Second-Order Accurate Total Variation Diminishing (TVD) Schemes to the Euler Equations in General Geometries.* NASA TM-85845, 1983.

[80] Yee, H.C.: *Upwind and Symmetric Shock-Capturing Schemes.* NASA TM-89464, 1987.

[81] Yee, H.C.; Harten, A.: *Implicit TVD Schemes for Hyperbolic Conservation Laws in Curvilinear Coordinates.* AIAA Journal, 25 (1987), pp. 266-274.

[82] Yee, H.C.; Klopfer, G.H.; Montagné, J.-L.: *High-Resolution Shock-Capturing Schemes for Inviscid and Viscous Hypersonic Flows.* NASA TM-100097, 1988.

[83] Yee, H.C.: *A Class of High-Resolution Explicit and Implicit Shock-Capturing Methods.* VKI Lecture Series 1989-04, 1989; also NASA TM-101088, 1989.

[84] Kroll, N.; Gaitonde, D.; Aftosmis, M.: *A Systematic Comparative Study of Several High Resolution Schemes for Complex Problems in High Speed Flows.* 29th AIAA Aerospace Sci. Meeting and Exhibit, Reno, USA, 1991.

[85] Kroll, N.; Rossow, C.-C.: *A High Resolution Cell Vertex TVD Scheme for the Solution of the Two- and Three-Dimensional Euler Equations.* 12th International Conf. on Numerical Methods in Fluid Dynamics, Oxford, UK, 1990.

[86] Müller, B.: *Simple Improvements of an Upwind TVD Scheme for Hypersonic Flow.* AIAA Paper 89-1977, 1989.

[87] Blazek, J.: *Methods to Accelerate the Solutions of the Euler- and Navier-Stokes Equations for Steady-State Super- and Hypersonic Flows.* Translation of DLR Research Report 94-35, ESA-TT-1331, 1995.

[88] LeVeque, R.J.: *Numerical Methods for Conservation Laws.* Birkhäuser Verlag, Basel, Switzerland, 1992.

[89] Spekreijse, S.P.: *Multigrid Solution of the Steady Euler Equations.* Ph.D. Dissertation, Centrum voor Wiskunde en Informatica, Amsterdam, The Netherlands, 1987.

[90] Venkatakrishnan, V.: *Preconditioned Conjugate Gradient Methods for the Compressible Navier-Stokes Equations.* AIAA Journal, 29 (1991), pp. 1092-1110.

[91] Van Albada, G.D.; Van Leer, B.; Roberts, W.W.: *A Comparative Study of Computational Methods in Cosmic Gas Dynamics.* Astronomy and Astrophysics, 108 (1982), pp. 76-84.

[92] Venkatakrishnan, V.: *On the Accuracy of Limiters and Convergence to Steady State Solutions.* AIAA Paper 93-0880, 1993.

[93] Hemker, P.W.; Koren, B.: *Multigrid, Defect Correction and Upwind Schemes for the Steady Navier-Stokes Equations.* Synopsis, HERMES Hypersonic Research Program Meeting, Stuttgart, Germany, 1987.

[94] Radespiel, R.; Swanson, R.C.: *An Investigation of Cell Centred and Cell Vertex Multigrid Schemes for the Navier-Stokes Equations.* AIAA Paper 89-0548, 1989.

[95] Radespiel, R.: *A Cell-Vertex Multigrid Method for the Navier-Stokes Equations.* NASA TM-101557, 1989.

[96] Swanson, R.C.; Radespiel, R.: *Cell Centered and Cell Vertex Multigrid Schemes for the Navier-Stokes Equations.* AIAA Journal, 29 (1991), pp. 697-703.

[97] Morton, K.W.; Paisley, M.F.: *On the Cell-Centre and Cell-Vertex Approaches to the Steady Euler Equations and the Use of Shock Fitting.* Proc. Int. Conf. Num. Meth. Fluid Dynamics 10, Beijing, China, 1986.

[98] Dimitriadis, K.P.; Leschziner, M.A.: *Multilevel Convergence Acceleration for Viscous and Turbulent Transonic Flows Computed with Cell-Vertex Method.* Proc. 4th Copper Mountain Conference on Multigrid Methods, Colorado, SIAM, pp. 130-148, 1989.

[99] Dick, E.: *A Flux-Vector Splitting Method for Steady Navier-Stokes Equations.* Int. J. Num. Meth. Fluids, 8 (1988), pp. 317-326.

[100] Dick, E.: *A Multigrid Method for Steady Incompressible Navier-Stokes Equations Based on Partial Flux Splitting.* Int. J. Num. Meth. Fluids, 9 (1989), pp. 113-120.

[101] Crumpton, P.I.; Mackenzie, J.A.; Morton, K.W.: *Cell Vertex Algorithms for the Compressible Navier-Stokes Equations.* J. Computational Physics, 109 (1993), pp. 1-15.

Chapter 5

Spatial Discretisation: Unstructured Finite Volume Schemes

As we already noted in the introduction to Chapter 3, the majority of numerical schemes for the solution of the Euler- and the Navier-Stokes equations employ the *method of lines*, i.e., a separate discretisation in space and time. The main advantage of this approach is that it allows us to select numerical approximations of different accuracy for the spatial and temporal derivatives. This offers a significantly larger flexibility as compared to methods based on *coupled* space and time discretisation, like the Lax-Wendroff family of schemes (e.g., explicit MacCormack predictor-corrector scheme, implicit Lerat's scheme, etc. – details may be found, e.g., in Ref. [1]). Because of the popularity of the method of lines, we shall follow this approach here.

The finite volume schemes which are discussed in this chapter are based on the conservation laws, as are represented by the Navier-Stokes (2.19) or by the Euler equations (2.45). In a pre-processing step, the physical domain is first subdivided into a number of elements (grid cells). In two dimensions, the elements are triangles, sometimes combined with quadrilaterals. In three dimensions, tetrahedra are most often employed [2]-[7]. However, an increasing number of flow solvers uses a mix of tetrahedra, prisms, pyramids, and in some cases also hexahedra (Fig. 5.1) for the simulation of high Reynolds number viscous flows [8]-[16]. Unstructured grids composed of various cell types are referred to as *mixed grids*. Examples are provided in Figs. 3.3 and 5.2. The designation 'mixed grids' should not be confused with the term *hybrid grids*, which means combined structured-unstructured grids (e.g., [17]-[19]).

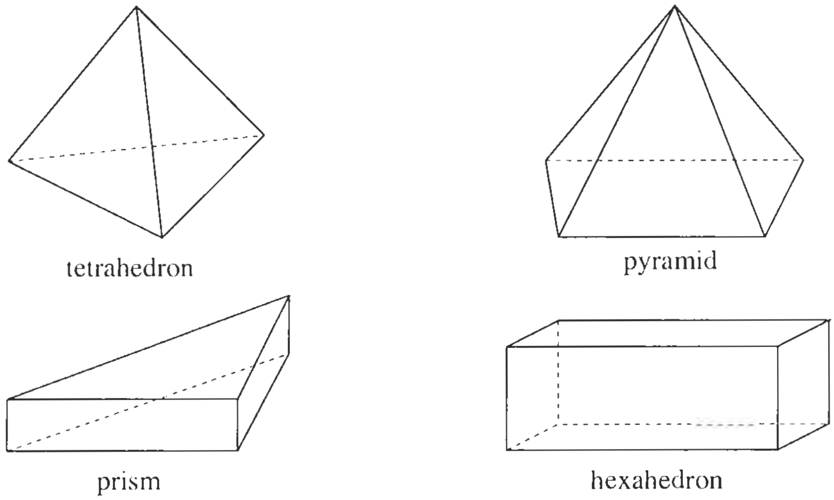


Figure 5.1: Elements used for the generation of 3-D unstructured grids.

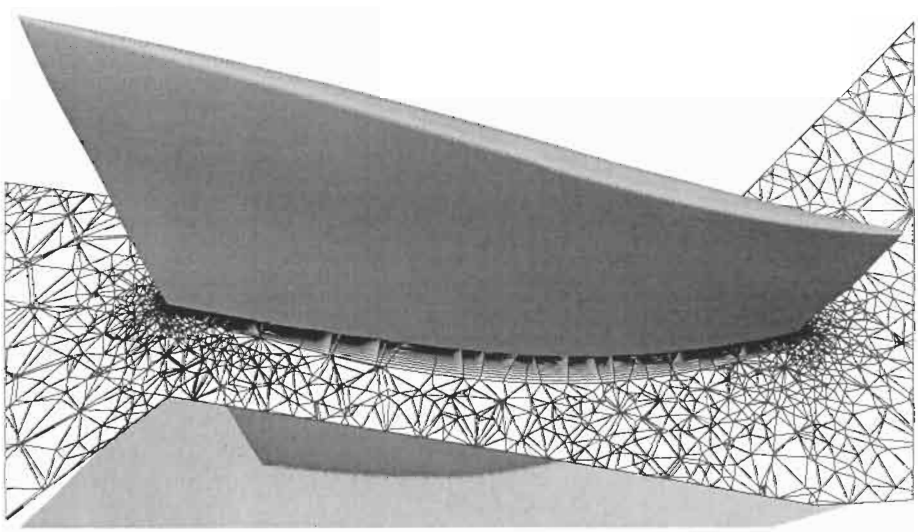


Figure 5.2: Planar cut through a 3-D unstructured mixed grid around a compressor blade. Grid was generated using CENTAUR^{TM} [20], [21]. Note the layers of quadrilateral faces around the surface which is due to the prisms. Irregularity of the tetrahedral grid is caused by the planar cut.

The grid generation has to be done in a way that preserves the conservation properties of the governing equations, namely:

- the physical domain has to be completely covered by the grid,
- there must be no free space left between the elements,
- the elements may not overlap.

In addition to fulfilling the above requirements, the grid should be smooth, i.e., there should be no large differences in the volumes or in the stretching ratio of adjacent grid cells and the elements should be as regular as possible. Otherwise, the numerical errors could spoil the solution accuracy completely [22], [23].

Based on the grid, suitable control volumes are defined in order to evaluate the integrals of the convective and viscous fluxes as well as of the source term. For simplicity, let us assume that a particular control volume does not change in time (otherwise see Appendix A.4). Then, the time derivative of the conservative variables $\vec{W}$ can be cast in the form

$$\frac{\partial}{\partial t}\int_{\Omega} \vec{W}\, d\Omega = \Omega\, \frac{\partial \vec{W}}{\partial t}\,.$$

Herewith, Eq. (2.19) becomes

$$\frac{\partial \vec{W}}{\partial t} = -\frac{1}{\Omega}\left[\oint_{\partial\Omega}(\vec{F}_c - \vec{F}_v)\, dS - \int_{\Omega}\vec{Q}\, d\Omega\right]\,. \tag{5.1}$$

The surface integral on the right-hand side of Equation (5.1) is approximated by a sum of the fluxes crossing the faces of the control volume. This approximation is called *spatial discretisation*. It is usually supposed that the flux is constant along the individual face and that it is evaluated at the midpoint of the face. This treatment is sufficient for a second-order accurate scheme. The source term is generally assumed to be constant inside the control volume. However, in cases where the source term becomes dominant, it is advisable to evaluate $\vec{Q}$ as the weighted sum of values from the neighbouring control volumes (see [24] and the references cited therein). If we consider a particular volume Ω_I, we obtain from Eq. (5.1)

$$\frac{d\vec{W}_I}{dt} = -\frac{1}{\Omega_I}\left[\sum_{m=1}^{N_F}(\vec{F}_c - \vec{F}_v)_m\, \Delta S_m \;-\; (\vec{Q}\Omega)_I\right]\,. \tag{5.2}$$

In the above expression, the index I in capital letters references the control volume, since in general it does not necessarily coincide with the grid, as we shall see later. Furthermore, N_F denotes the number of the faces of the control volume Ω_I, and the variable ΔS_m stands for the area of the face m, respectively. The number of faces N_F depends of course on the cell-type but also on the type of the control volume. In general, the number of faces changes between the control volumes as well, which is one of the main differences as compared to structured grids. However, numerical procedures and data structures were

developed which avoid the a priori knowledge of N_F. We shall return to this point in later sections.

The term in square brackets on the right-hand side of Eq. (5.2) is usually denoted as the *residual*. Thus, we may abbreviate Eq. (5.2) as

$$\frac{d\vec{W}_I}{dt} = -\frac{1}{\Omega_I}\vec{R}_I\,. \tag{5.3}$$

Writing down the relationship in Equation (5.3) for all control volumes Ω_I, we obtain a system of ordinary differential equations of first order. The equations are hyperbolic in time, that means we have to advance them in time starting from a known initial solution. We have also to provide appropriate boundary conditions for the viscous and the inviscid fluxes, as they are described in Chapter 8.

When numerically solving the system of discretised governing equations (5.3), the first question is how to define the control volumes and where to locate the flow variables with respect to the grid points. In the framework of finite volume schemes, three basic strategies can be pursued:

- *Cell-centred* scheme [25], [26], [2], [16] – control volumes are identical with the grid cells and the flow variables are associated with their centroids (Fig. 5.6).

- *Cell-vertex* scheme with *overlapping* control volumes [27], [28] – flow quantities are assigned to the grid vertex and the control volumes are defined as the union of all grid cells having the respective node in common. This means that the control volumes associated with two neighbouring vertices overlap each other.

- Cell-vertex scheme with *median-dual* control volumes [29]-[33], [13], [15] – flow variables are again stored at the grid vertices, but the control volumes are now created by connecting the centroids of the surrounding elements, face-centroids and edge-midpoints (Fig. 5.8). In this way, the grid points are encapsulated by their corresponding control volumes – representing a *dual grid* – which do not overlap.

Because the cell-vertex scheme with overlapping control volumes is no longer used, we shall concentrate here on the cell-centred and on the median-dual scheme. Both methodologies will be discussed in detail in Section 5.2.

It is important to notice that in our case **all** flow variables, i.e., the conservative variables (ρ, ρu, ρv, ρw and ρE) and the dependent variables (p, T, c, etc.), are associated with the **same** location – with the cell centre or with the grid point. This approach is known as the *co-located* grid scheme. By contrast, many older (structured) pressure-based methods (cf. Section 3.1) use the so-called *staggered* grid scheme, where the pressure and the velocity components are stored at different locations in order to suppress oscillations of the solution which arise from central differencing.

Many choices exist with respect to the evaluation of the convective fluxes. The basic problem is that we have to know their values at all N_F faces of a control volume, but the flow variables are not directly available there. This means, we have to interpolate either the fluxes or the flow variables to the faces of the control volume. The interpolation of flow variables is known as *reconstruction* of the solution from values inside the control volumes (see Subsection 5.3.3). In principle, the interpolation can be conducted in one of two ways:

- by arithmetic averaging like in *central* discretisation schemes;
- by some biased interpolation like in *upwind* discretisation schemes, which take care of the characteristics of the flow equations.

Besides the description, we shall treat aspects such as accuracy, range of applicability and numerical effort of the most widely used discretisation schemes for the convective fluxes in Section 5.3.

A commonly applied methodology for the evaluation of the viscous fluxes at a face of the control volume is based on arithmetic averaging of the flow quantities. More involved is the calculation of the velocity and the temperature gradients in Equations (2.15) and (2.24), particularly in the case of mixed grids. We shall present the whole procedure in Section 5.4.

5.1 Geometrical Quantities of a Control Volume

Before we start to discuss the discretisation methodologies applied to the convective and viscous fluxes, it is important to consider the evaluation of geometrical quantities of the control volume Ω_I – its volume, the unit normal vector $\vec{n}_m$ (defined as outward facing) and the area ΔS_m of a face m. The normal vector and the face area are also denoted as the *metrics* of the control volume. In the following, we shall consider the 2-D and the 3-D case separately.

5.1.1 Two-Dimensional Case

Generally, we think of the flow in a plane as being a special case of a 3-D problem, where the solution is symmetric with respect to one coordinate direction (e.g., to the z-direction). Because of the symmetry and in order to obtain correct physical units for volume, pressure, etc., we set the depth of all grid cells and control volumes equal to a constant value b. The volume of a control volume results then in 2D from the product of its area with the depth b. Since the depth b is arbitrary, we may set $b = 1$ for convenience. In the following discussion, we restrict ourselves to triangular and quadrilateral elements. Even though the control volume of a median-dual scheme can have a rather complex shape, it can always be decomposed into triangles and/or quadrilaterals.

Triangular element

The area of a general triangle can be most conveniently and exactly calculated by the formula of Gauss. Thus, using a node numbering in accordance with Fig. 5.3a, the volume results from

$$\begin{aligned} \Omega = \frac{b}{2} \big[& (x_1 - x_2)(y_1 + y_2) \\ & + (x_2 - x_3)(y_2 + y_3) \\ & + (x_3 - x_1)(y_3 + y_1) \big] . \end{aligned} \tag{5.4}$$

The nodes have to be numbered in the anti-clockwise direction in order to obtain a positive value for the volume.

Quadrilateral element

The area of a general quadrilateral can be exactly calculated by Gauss' formula, which leads, after some algebra, to the expression

$$\Omega = \frac{b}{2} \left[(x_1 - x_3)(y_2 - y_4) + (x_4 - x_2)(y_1 - y_3) \right] , \tag{5.5}$$

where the nodes are numbered according to Fig. 5.3b in the anti-clockwise direction. In the above, we assumed that the control volume is located in the x-y-plane and that the z-coordinate represents the symmetry axis.

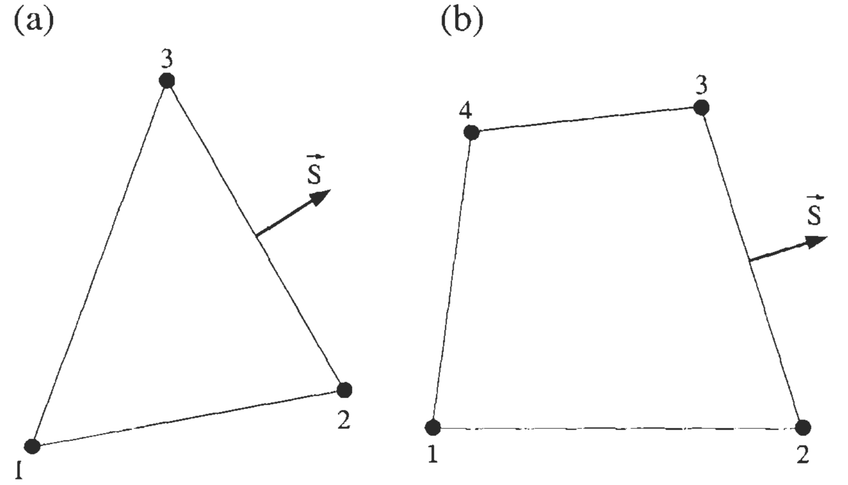


Figure 5.3: Numbering of nodes and face vector of: (a) triangular element, (b) quadrilateral element.

The edges of a control volume are given by straight lines in 2D and therefore the unit normal vector is constant along them. When we integrate the fluxes according to the approximation of Eq. (5.2), we have to evaluate the product of the area of a face ΔS and the corresponding unit normal vector $\vec{n}$ which is the *face vector* $\vec{S}$. Considering Fig. 5.3, the outward pointing face vector, e.g., at the side 2-3 is given by

$$\vec{S}_{23} = \vec{n}_{23}\,\Delta S_{23} = b\begin{bmatrix} y_3 - y_2 \\ x_2 - x_3 \end{bmatrix} . \tag{5.6}$$

Because of the symmetry, the z-component of the face vectors (and the unit normal vector) is zero. It is therefore omitted in Eq. (5.6). The unit normal vector can be obtained from Eq. (5.6) with

$$\Delta S = |\,\vec{S}\,| = \sqrt{S_x^2 + S_y^2}\,, \tag{5.7}$$

where S_x, S_y denote the Cartesian components of the face vector.

5.1.2 Three-Dimensional Case

As opposed to the previous 2-D case, the computation of face vectors and volumes poses in 3D some problems for elements or control volumes with quadrilateral faces. The main reason for this is that, in general, the four vertices of a quadrilateral face of a control volume may not lie in a plane. Then, the normal vector is no longer constant on such face (see Fig. 4.2). In order to overcome this difficulty, we could decompose each quadrilateral face into two or even more

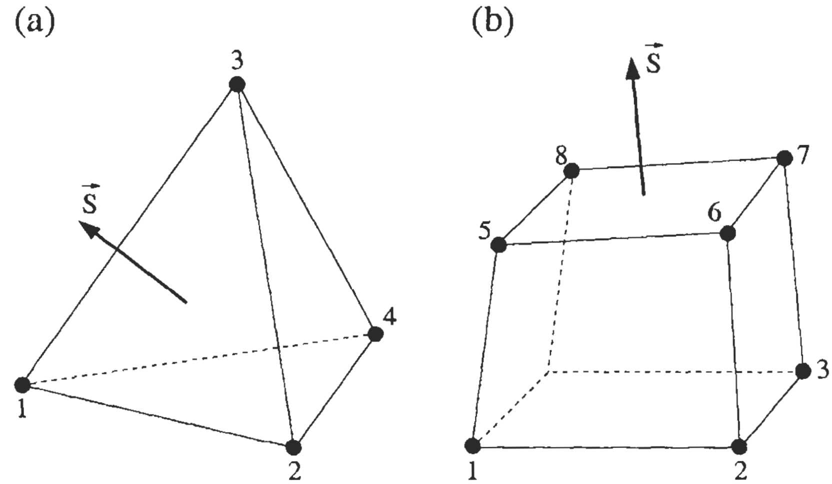


Figure 5.4: Numbering of nodes and face vector of: (a) tetrahedral element, (b) hexahedral element.

triangles. However, the gain in accuracy is hardly noticeable for a second-order scheme on a smooth grid. The additional effort can only be justified – and in fact it becomes necessary – for a third- and higher order spatial discretisations. Therefore, we shall apply a simplified treatment of the quadrilateral faces in the following considerations, which is based on an averaged normal vector.

Triangular face

The face vector $\vec{S}$ can be exactly computed for a triangular face using Gauss' formula. Defining the nodes according to Fig. 5.4a, we obtain for the edge differences of the triangle 1-2-3

$$\begin{aligned}
\Delta xy_A &= (x_1 - x_2)(y_1 + y_2), \quad \Delta yz_A = (y_1 - y_2)(z_1 + z_2), \\
\Delta xy_B &= (x_2 - x_3)(y_2 + y_3), \quad \Delta yz_B = (y_2 - y_3)(z_2 + z_3), \\
\Delta xy_C &= (x_3 - x_1)(y_3 + y_1), \quad \Delta yz_C = (y_3 - y_1)(z_3 + z_1), \\
\Delta zx_A &= (z_1 - z_2)(x_1 + x_2), \\
\Delta zx_B &= (z_2 - z_3)(x_2 + x_3), \\
\Delta zx_C &= (z_3 - z_1)(x_3 + x_1)\,.
\end{aligned} \tag{5.8}$$

The outward pointing face vector $\vec{S} = \vec{n}\Delta S$ results then from

$$\vec{S} = \frac{1}{2}\begin{bmatrix} \Delta yz_A + \Delta yz_B + \Delta yz_C \\ \Delta zx_A + \Delta zx_B + \Delta zx_C \\ \Delta xy_A + \Delta xy_B + \Delta xy_C \end{bmatrix}. \tag{5.9}$$

Quadrilateral face

The averaged face vector $\vec{S}$ of a quadrilateral face, like that rendered in Fig. 5.4b, is most conveniently computed using the same Gauss' formula as employed in 2-D for the area of a quadrilateral. Thus, for the face given by the nodes 5, 6, 7 and 8 in Fig. 5.4b, we first define the differences

$$\begin{aligned} \Delta x_A &= x_8 - x_6\,, & \Delta x_B &= x_7 - x_5\,, \\ \Delta y_A &= y_8 - y_6\,, & \Delta y_B &= y_7 - y_5\,, \\ \Delta z_A &= z_8 - z_6\,, & \Delta z_B &= z_7 - z_5\,. \end{aligned} \tag{5.10}$$

Then, we obtain the outward pointing face vector $\vec{S} = \vec{n}\Delta S$ from the relation

$$\vec{S} = \frac{1}{2}\begin{bmatrix} \Delta y_A\,\Delta z_B - \Delta z_A\,\Delta y_B \\ \Delta z_A\,\Delta x_B - \Delta x_A\,\Delta z_B \\ \Delta x_A\,\Delta y_B - \Delta y_A\,\Delta x_B \end{bmatrix}. \tag{5.11}$$

The approximation becomes exact when the face approaches a parallelogram, i.e., when the vertices of the face lie all in one plane.

The unit normal vector is obtained in both cases from $\vec{n} = \vec{S}/\Delta S$ with

$$\Delta S = \sqrt{S_x^2 + S_y^2 + S_z^2}\,, \tag{5.12}$$

where S_x, S_y and S_z denote the Cartesian components of the face vector given by Eq. (5.9) or Eq. (5.11), respectively.

Volume

As we already stated in the case of 3-D structured finite volume schemes, a very convenient approach for the computation of volumes is based on the divergence theorem [34]. The discussion in Subsection 4.1.2 led finally to the expression

$$\Omega = \frac{1}{3}\sum_{m=1}^{N_F}(\vec{r}_{\text{mid}}\cdot\vec{S})_m \tag{5.13}$$

for the volume, where N_F denotes the number of the faces of the control volume, $(\vec{r}_{\text{mid}})_m$ the the midpoint of the control volume face m, and $\vec{S}_m$ the face vector (outward directed) at face m, respectively. The formula (5.13) is directly applicable on unstructured grids. It is exact for a volume with triangular faces, or a volume with planar quadrilateral faces.

5.2 General Discretisation Methodologies

We already mentioned at the beginning of this chapter that there are two popular approaches for the definition of the control volume and for the location of the flow variables. These are the cell-centred scheme and the median-dual scheme. We shall present both in more detail in this section.

However, before we start, let us say a few words about the basic data structure which is needed for an unstructured flow solver. In fact, a flexible but in terms of memory and operation count efficient data structure is the crucial point of any unstructured scheme. You can say that the structure which is missing in the grid has to be provided inside the solver. At least the following data is required:

- coordinates of the grid nodes (vertices),
- pointers from elements to grid nodes,
- pointers from faces of elements located on a boundary to grid nodes.

Further data structures, which are required by the discretisation schemes, can be generated from this information. In order to illustrate how the above data could possibly be stored, let us consider for example the tetrahedron in Fig. 5.4a. If we further assume that the face 1-2-4 is on a boundary (wall, inlet, farfield, etc.), we could employ the format:

```
# nodes (x, y, z):
  P1.x  P1.y  P1.z
  P2.x  P2.y  P2.z
  P3.x  P3.y  P3.z
  P4.x  P4.y  P4.z
  ...
# tetrahedra:
  ...
  P1  P2  P3  P4
  ...
# boundaries:
  ...
  type  P1  P4  P2
  ...
```

A similar format is also utilised by the 2-D unstructured code provided on the accompanying CD-ROM.

It is important to realise the following two points related to the boundaries of the computational domain. First, it is more convenient to store boundary faces than just the nodes. This can be understood by considering the situation depicted in Fig. 5.5. The problem is that node P_1 is shared by three, nodes P_2 and P_3 by two boundaries of possibly physically different types. Therefore, it can become very cumbersome to apply the correct boundary conditions. On contrary, a face can belong to only **one** boundary, like P_1-P_2-P_4 to boundary 1.

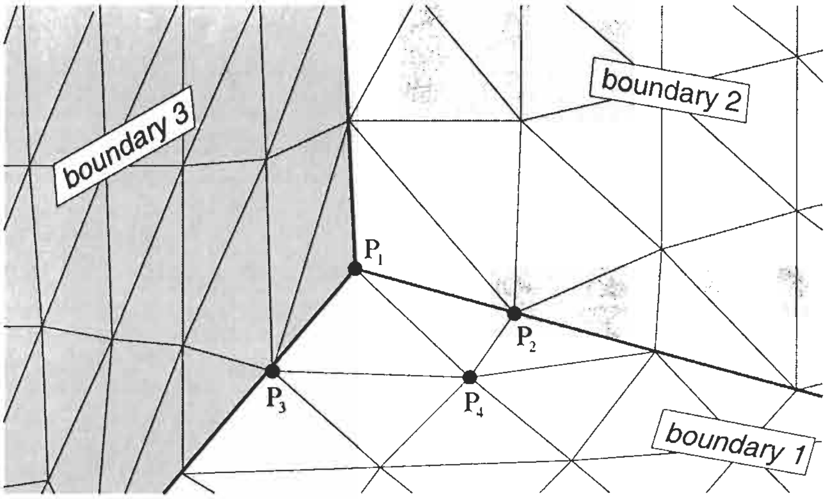


Figure 5.5: Three boundaries which meet at one corner – ambiguity of grid points with respect to boundary type.

The second point concerns the numbering of the nodes of the boundary faces. This has to be done in a consistent way – e.g., anti-clockwise when viewed from outside the flow domain – in order to have all face vectors (Eqs. (5.7), (5.9) or (5.11)) either pointing outward or inward.

5.2.1 Cell-Centred Scheme

We speak of a cell-centred scheme if the control volumes are identical with the grid cells and if the flow variables are associated with the centres of the grid cells as sketched in Fig. 5.6. When we evaluate the discretised flow equations (5.2), we have to supply the convective and the viscous fluxes at the midpoints of the faces of the control volume, which is sufficient for a second-order accurate discretisation on smooth grids (averaged normal vector employed for quadrilateral faces). The fluxes can be approximated in one of three ways:

1. by the *average of fluxes* computed from values at the centres of the grid cells to the left and to the right of the cell face, but using the same unit normal vector (generally applied only to the convective fluxes);

2. by using an *average of variables* associated with the centres of the grid cells adjacent to the left and to the right side of the cell face;

3. by computing the fluxes from flow quantities *reconstructed* separately on both sides of the cell face from values in the surrounding cells (employed only for the convective fluxes).

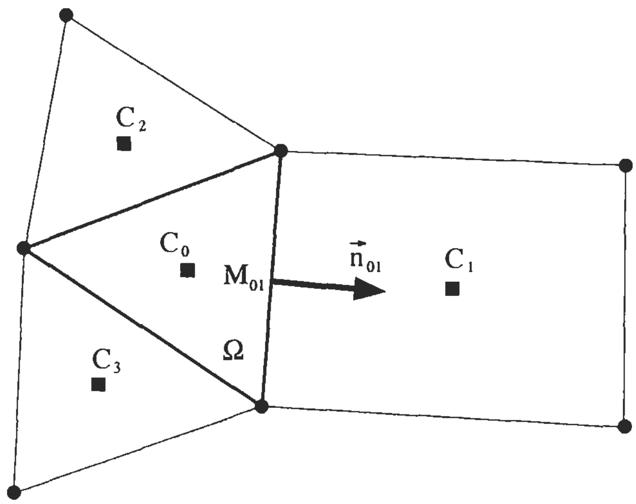


Figure 5.6: Control volume of a cell-centred scheme (in 2D). Grid nodes are represented by circles, cell centres by rectangles (C).

Thus, considering for example the cell face with the unit normal vector $\vec{n}_{01}$ in Fig. 5.6, the first approach – average of fluxes – reads in two dimensions

$$(\vec{F}_c \, \Delta S)_{01} \approx \frac{1}{2} \left[\vec{F}_c(\vec{W}_0, \vec{n}_{01}) + \vec{F}_c(\vec{W}_1, \vec{n}_{01}) \right] \Delta S_{01} \tag{5.14}$$

with the face area ΔS_{01} computed from Eqs. (5.6) and (5.7).

The second approach – average of variables – can be formulated as follows

$$(\vec{F} \, \Delta S)_{01} \approx \vec{F}(\vec{W}_{01}, \vec{n}_{01}) \, \Delta S_{01} \,, \tag{5.15}$$

where the conservative/dependent variables at the face with the unit normal vector $\vec{n}_{01}$ are defined as the arithmetic average of values at the two adjacent cells. Thus,

$$\vec{W}_{01} = \frac{1}{2} \left(\vec{W}_0 + \vec{W}_1 \right) . \tag{5.16}$$

The flux vector $\vec{F}$ in Eq. (5.15) represents either the convective or the viscous fluxes.

The third methodology starts with an interpolation of flow quantities (usually velocity components, pressure, density and total enthalpy) separately to both sides of the cell face. The reconstructed quantities – termed the *left* and the *right state* (see Subsection 5.3.3) – differ in general. The fluxes through the cell face are then evaluated from the difference of the left and right state using some non-linear function. Hence,

$$(\vec{F}_c \, \Delta S)_{01} \approx f_{Flux} \left(\vec{U}_L, \vec{U}_R, \Delta S_{01} \right) , \tag{5.17}$$

where

$$
\begin{aligned}
\vec{U}_L &= f_{Rec}\left(\cdots, \vec{U}_2, \vec{U}_0, \cdots\right) \\
\vec{U}_R &= f_{Rec}\left(\cdots, \vec{U}_1, \vec{U}_0, \cdots\right)
\end{aligned}
\tag{5.18}
$$

represent the reconstructed states.

Of course, similar relations hold for the other control volume faces as well. The above approximations can be employed in the same way in three dimensions. The face vector $\vec{S}$ is then evaluated using the formulae (5.9) or (5.11), respectively.

As we already stated in the introduction to this section, the basic data structure which describes the elements has to be extended in an appropriate way to support the discretisation methodology. It is obvious from the previous discussion that numerical operations are carried out using mainly the faces of the elements (control volumes) together with values at the centres of the adjacent cells. It is therefore quite natural to employ a face-based data structure for the spatial discretisation. Such data structure stores for each particular face in the grid:

- pointers to the two cells which share the respective face – this allows it to access the flow variables associated with the two cells (C_0, C_1);
- the face vector ($\vec{S}_{01} = \vec{n}_{01}\Delta S_{01}$) – must point consistently either outwards or inwards;
- two vectors from each cell centre to the midpoint of the face M_{01}, i.e., (C_0-M_{01}), (C_1-M_{01}) – required for accurate interpolation of flow variables to the face. This is not necessary for purely tetrahedral grids, where a simple extrapolation formula can be used [26], [2] (see Eq. (5.40)).

Hence, the integration of the fluxes (e.g., according to Eq. (5.15)) would be implemented as a loop over **all** (i.e., internal and boundary) faces contained in the grid:

DO *face* = 1, *nfaces*

I = pointer_to_left_cell(*face*)

J = pointer_to_right_cell(*face*)

$\left(\vec{F}\,\Delta S\right)_{IJ} \approx \vec{F}(\vec{W}_{IJ}, \vec{n}_{IJ})\,\Delta S_{IJ}$

$\vec{R}_I = \vec{R}_I + \left(\vec{F}\,\Delta S\right)_{IJ}$

$\vec{R}_J = \vec{R}_J - \left(\vec{F}\,\Delta S\right)_{IJ}$

ENDDO

After the loop is completed and the source term $\vec{Q}_I\Omega_I$ is added, we obtain the final residuals ($\vec{R}$) in all cells. A less efficient approach would be to loop

over elements because the face vectors would have to be stored twice and the fluxes would be computed twice (with the exception of boundaries). Furthermore, because we use exactly the same face vector $\vec{S}_{IJ}$ in order to to evaluate the partial fluxes into the volumes Ω_I and Ω_J, the conservation properties of the governing equations are kept.

5.2.2 Median-Dual Cell-Vertex Scheme

Within the cell-vertex scheme, the flow variables are associated with the grid nodes (vertices). Median-dual control volumes are formed by connecting the centroids, face- and edge-midpoints of all cells sharing the particular node. This is depicted in Fig. 5.7a for a tetrahedron and in Fig. 5.7b for a hexahedron. The definition of a median-dual control volume results in a polyhedral hull around each grid node, as it is sketched in Fig. 5.8 for a 2-D mixed grid. This polyhedra can be viewed as a *dual* grid – hence the name of the scheme. It is interesting to note that the median-dual finite volume discretisation is equivalent to the Galerkin finite element scheme with linear elements (see, e.g., [35]).

In order to evaluate the discretised flow equations (5.2), we have to integrate the convective and viscous fluxes over the surface of the control volume. Hence, we would have to compute the fluxes for each partial face (e.g., F_1-M_{13}-F_2-C in Fig. 5.7a) separately. However, this is only required for a third- or higher-order accurate discretisations [36], [37]. In the case of a second-order scheme, which is most frequently employed, we may assume the flow variables to be constant for all faces grouped around a particular edge. The fluxes are then evaluated at the midpoint of the edge using the variables and the gradients from both nodes. This approach allows us to define a mean unit normal vector and a total face area associated with each edge. Thus referring to Fig. 5.8, the mean normal vector, e.g., for the edge P_0-P_1 becomes

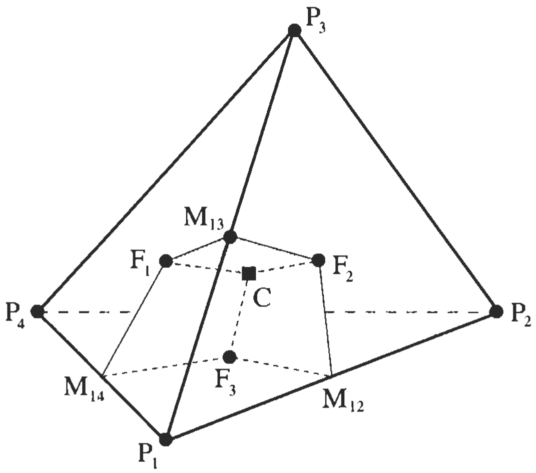
$$\vec{n}_{01} = \vec{n}_L + \vec{n}_R \,, \tag{5.19}$$

and the total face area is given by: $\Delta S_{01} = \Delta S_L + \Delta S_R$. The same applies also in 3D, where the mean normal vector results from a sum over all partial faces having the particular edge-midpoint in common, as it is rendered in Fig. 5.9. The face vector ($\vec{S} = \vec{n}\,\Delta S$) is computed in 2D from Eq. (5.6). In three dimensions, where the partial faces are always quadrilaterals, we can either divide them into triangles and use Eq. (5.9), or we can employ a simplified treatment due to Eq. (5.11), which is sufficient for smooth grids.

The fluxes can then be evaluated according to one of the three following methodologies:

1. by the *average of fluxes* computed from values at both nodes of an edge, but using the same mean unit normal vector (generally applied only to the convective fluxes);

(a)



(b)

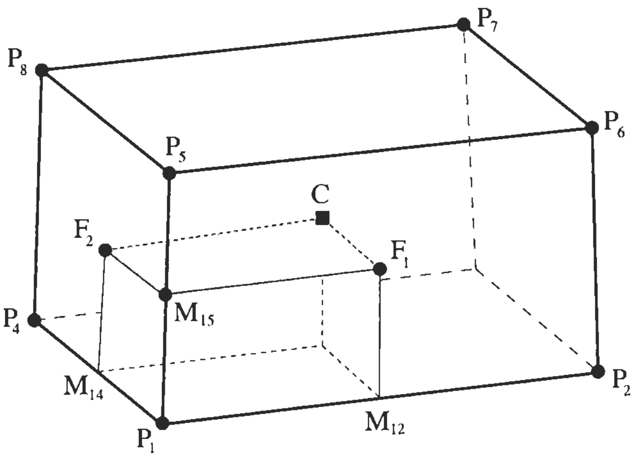


Figure 5.7: Partial control volume and faces (shaded) of a median-dual scheme for tetrahedron (a) and hexahedron (b). P denotes grid nodes, C cell-centroids, F face-centroids, and M stands for edge-midpoints. Shaded area represents one part of the control volume face assigned to edge P_1-P_3 or P_1-P_5, respectively.

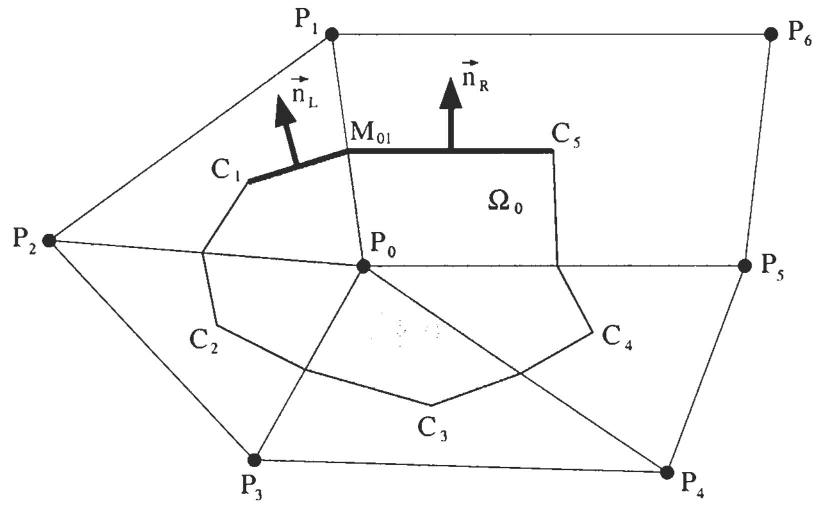


Figure 5.8: Control volume of a median-dual scheme (in 2D). C_1, C_2, etc. denote cell centres; P_1, P_2, etc. represent grid nodes. Face area associated with edge P_0-P_1 is rendered by bold line.

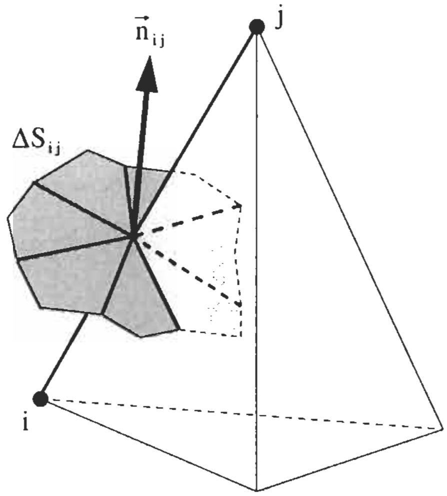


Figure 5.9: Total face area and mean unit normal vector associated with some edge ij of the 3-D median-dual cell-vertex scheme.

2. by using an *average of variables* stored at the two nodes of an edge;

3. by computing the fluxes from flow quantities *reconstructed* separately on both sides of the face of the control volume from values at the surrounding nodes (employed only for the convective fluxes).

The computation of fluxes follows formally the same approaches as for the cell-centred scheme. Thus, the formulae (5.14)-(5.18) are applicable also in the case of the median-dual scheme. If we utilise the above approach which associates each edge with a mean unit normal, the most efficient method is to employ an edge-based data structure for the spatial discretisation. The edge-based data structure stores for each particular edge in the grid (cf. Fig. 5.9):

- pointers to the two nodes which define the edge – this allows it to access the flow variables associated with the two control volumes Ω_i and Ω_j;

- the face vector ($\vec{S}_{ij} = \vec{n}_{ij}\Delta S_{ij}$) – must point consistently either outwards or inwards;

- the edge vector from node i to node j – required for the interpolation of flow variables to the face (solution reconstruction). Alternatively, the edge vector can be computed on the fly from coordinates of the nodes. This is not required for the standard central scheme with artificial dissipation (Subsection 5.3.1).

With this, the integration of the fluxes (e.g., according to Eq. (5.15)) would be implemented as a loop over **all** edges in the grid:

DO *edge* = 1, *nedges*

i = pointer_to_left_node(*edge*)

j = pointer_to_right_node(*edge*)

$$(\vec{F}\,\Delta S)_{ij} \approx \vec{F}(\vec{W}_{ij}, \vec{n}_{ij})\,\Delta S_{ij}$$

$$\vec{R}_i = \vec{R}_i + (\vec{F}\,\Delta S)_{ij}$$

$$\vec{R}_j = \vec{R}_j - (\vec{F}\,\Delta S)_{ij}$$

ENDDO

After the loop is completed and the source term $\vec{Q}_i\Omega_i$ is added, we obtain the final residuals ($\vec{R}$) in all nodes. This approach is significantly more efficient than summing up the fluxes over each control volume separately, because we store each mean face vector only once and we also visit each edge only once instead of twice. Furthermore, since we use exactly the same mean face vector $\vec{S}_{ij}$ in order to to evaluate the partial fluxes into the volumes Ω_i and Ω_j, mass, momentum and energy remain conserved.

5.2.3 Cell-Centred versus Median-Dual Scheme

The relative advantages and disadvantages of the cell-centred and the median-dual scheme are the subject of controversial debates. The main reason is the lack of fair comparisons of the two methodologies with respect to accuracy, computational time and memory for realistic configurations. Our intention here is to collect the most important arguments for and against each of the approaches regarding:

- accuracy,
- computational work,
- memory requirements, and
- flexibility.

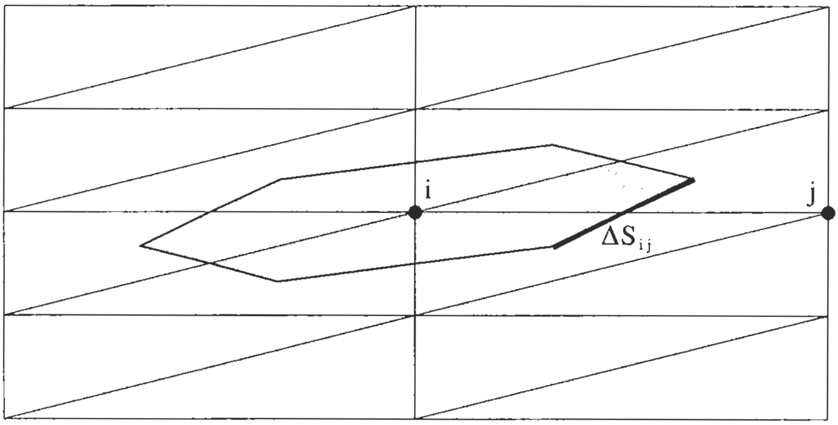
This should lead to a greater understanding of the problems inherent to each scheme and should be of help in selecting the most suitable scheme for the intended applications.

Accuracy

A cell-centred scheme on a triangular/tetrahedral grid leads to about twice/six times as many control volumes and hence degrees of freedom as a median-dual scheme [35]. On typical mixed grids, which consist of tetrahedra and prisms, a cell-centred scheme gives roughly three times more unknowns than a median-dual scheme. This suggests that cell-centred schemes are more accurate than cell-vertex discretisations on an identical grid. However, the residual of a cell-centred scheme results from a much smaller number of fluxes as compared to a median-dual scheme (three versus approximately seven on a tetrahedral grid), which may impair the accuracy. Thus, there is no clear evidence about which scheme might be superior.

The median-dual scheme suffers from a particular problem on stretched triangular and tetrahedral grids. Consider, for example, Fig. 5.10, which shows a tessellation composed of right triangles, as it is often employed near solid walls for viscous flows. We can see in Fig. 5.10a that the face ΔS_{ij} becomes highly skewed with respect to the edge ij. However, spatial discretisation schemes mostly assume fluxes to be orthogonal to a face (especially Riemann solvers). Thus, an error is introduced which is particularly significant for a first-order scheme [37]. The situation can be improved using the so-called *containment-dual* control volume [38]. As depicted in Fig. 5.11, the containment-dual approach employs the centres of the minimum spanning circles/spheres instead of cell-centroids to define the faces. This leads to control volumes identical to those on quadrilateral grids (Fig. 5.10b). Notice that there is no face area associated with diagonal edges like ij'. An additional effort is required for pre-processing, but the solution accuracy can be improved noticeably [39]. Of course, another possibility is to employ directly quadrilateral or hexahedral cells in boundary

(a)



(b)

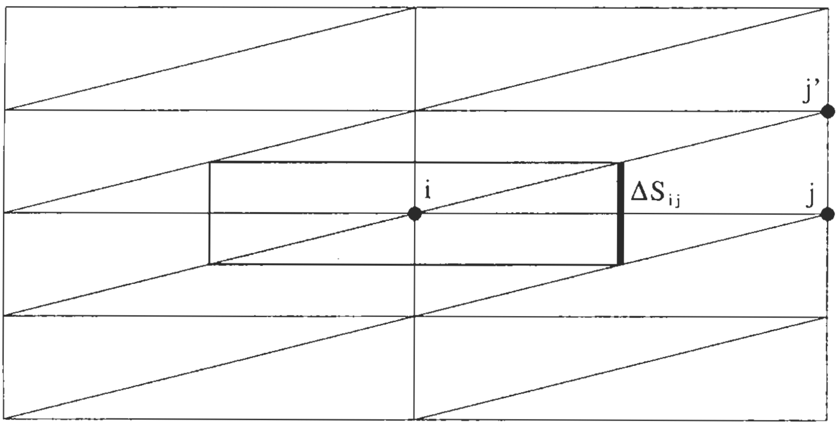


Figure 5.10: Comparison of median-dual (a) and containment-dual (b) control volumes for stretched right triangulation.

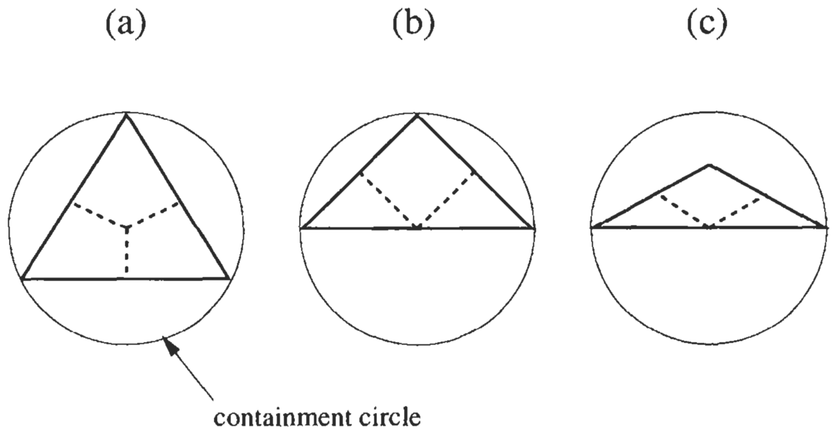


Figure 5.11: Part of containment dual (dashed line) in the case of acute (a) and obtuse (c) triangles [39]. The containment circle is the smallest circle which contains the triangle. For obtuse triangles, it is centred on the longest edge.

layers. Further discussion of grid-induced errors can be found in Ref. [22] and [23].

Another problem inherent to the median-dual scheme is the discretisation at boundaries of the physical domain. What happens is that there is only about one half of the control volume left at the boundary (cf. Fig. 4.6). The integration of fluxes around the faces results in a residual located **inside** – ideally at the **centre** – of the control volume. However, the residual is associated with the **node**, residing directly on the boundary. This mismatch leads to increased discretisation error in comparison to the cell-centred scheme, which is particularly undesirable on solid walls. The definition of the dual control volume causes also problems at sharp corners (like trailing edges), which show up as unphysical peaks in pressure or density. Further complications arise at periodic boundaries (see Chapter 8.7), where the fluxes from both parts of the control volume have to be summed up correctly.

The mismatch between the centre of the control volume and the node where the residual is stored has also a further implication for the median-dual scheme. It arises as the mass matrix in the case of unsteady flows. We discussed this point already at the beginning of Section 3.2. The advantage of the cell-centred scheme is that the mass matrix can be eliminated from the equations, without compromising the solution accuracy. By contrast, the median-dual scheme requires a special treatment of the mass matrix [40], [41].

Computational Work

In order to judge the computational effort required for both schemes, we have to consider primarily the integration of the fluxes. We know from the previous discussion that the cell-centred scheme uses a loop over cell faces whereas the median-dual scheme loops over edges. Since the evaluation of the fluxes at an interface is quite similar for both schemes, the ratio of the number of cell faces to the number of edges gives the ratio of the computational work. Thus, on a tetrahedral grid, where the number cell faces (if counted only once for each two cells) is approximately two times larger than the number of edges, the cell-centred scheme is computationally twice as much expensive as the median-dual scheme on an identical grid [35]. The cell-centred approach becomes however more competitive on mixed grids containing prismatic elements. Apart from boundary treatment, both methods are computationally equivalent on hexahedral grids, where the number of faces equals the number of edges.

Memory Requirements

Considering the memory requirements, the cell-centred scheme has to store about six times more flow variables on tetrahedral and about three times more variables on usual mixed grids as compared to the median-dual scheme. Furthermore, as we saw, both schemes require to store two integers and three reals (pointers and face vector) per cell face or edge, respectively. Additionally, the cell-centred scheme has to keep two vectors to the face-midpoint – 6 reals – per cell face in memory. On contrary, the median-dual scheme can work with the node coordinates only, which are considerably fewer values. Thus in summary, the cell-centred scheme needs, on average, more than twice as much computer memory as the median-dual method.

Grid Generation/Adaptation

One significant advantage of the cell-centred scheme appears in the case of non-conforming cell interfaces, like those at the letter "F" in Fig. 3.4. In contrast to the median-dual methodology, no special and expensive procedure is required for the computation of the fluxes at the interface. This allows for an increased flexibility in the grid generation and also in the grid adaptation.

5.3 Discretisation of Convective Fluxes

In the previous sections, we considered general issues of possible spatial discretisation methodologies including the necessary data structures. In what follows, we shall learn more about the details, how the evaluation of the convective fluxes can be implemented.

As we could already see in Subsection 3.1.5, in the framework of the finite volume approach, we have basically the choice between:

- central,
- flux-vector splitting,
- flux-difference splitting,
- total variation diminishing (TVD), and
- fluctuation-splitting

schemes. First of all, we shall present the central discretisation on unstructured grids at some length, since it differs considerably from that on structured grids. On contrary, the basics of the upwind schemes are identical on structured and unstructured grids. Hence, the details can be found in Sections 4.3.2-4.3.4. However, what is new on unstructured grids is the solution *reconstruction*, which is required in order to obtain the values of the flow variables at a face of the control volume. Therefore, we shall discuss the common approaches in some detail in Subsection 5.3.3. Because of space limitations, we will not treat the fluctuation-splitting approach here, which is still in research status. The reader is referred to Subsection 3.1.5 for the bibliography related to fluctuation-splitting schemes.

5.3.1 Central Schemes with Artificial Dissipation

The basic idea of the central scheme is to compute the convective fluxes at a face of the control volume from the arithmetic average of the conservative variables on both sides of the face according to Eq. (5.16). Since this would lead to odd-even decoupling of the solution (generation of two independent solutions of the discretised equations) and wiggles at shocks, artificial dissipation has to be added for stability. The artificial dissipation is based on a blend of second- and fourth-order differences. The scheme was first implemented for the Euler equations on structured grids by Jameson et al. [42]. Because of the names of the authors, it is also abbreviated as the JST scheme.

The implementation of the JST scheme on unstructured grids utilises the Laplacian operator for the second-order differences and the Laplacian of Laplacian for the fourth-order differences [43], [27]. In order to reduce the computational cost, pseudo-Laplacians are employed instead of true Laplacians. For this purpose, a 2-D formulation was proposed first in [44] and then improved in [45]. Later on, the scheme was extended to 3D in [2]. It makes use of a

distance-weighting procedure. In this way, the scheme results in the pseudo-Laplacian being zero for a linearly varying function on any grid. Applied to a general scalar quantity U in cell I, the pseudo-Laplacian takes the form

$$L(U_I) = \sum_{J=1}^{N_A} \theta_{IJ}\,(U_J - U_I)\,, \tag{5.20}$$

where N_A stands for the number of adjacent control volumes. The cell indices have to be substituted by node indices (i, j) in the case of the median-dual scheme. The sum in Eq. (5.20) is best evaluated using either a loop over faces (cell-centred scheme) or a loop over edges (median-dual scheme) similar to the flux computation. The geometrical weights θ are defined as

$$\theta_{IJ} = 1 + \Delta\theta_{IJ} \tag{5.21}$$

and result from the solution of an optimisation problem [2]. The optimisation problem is solved by means of Lagrange multipliers. Herewith, the the geometrical weights are obtained from the expression

$$\Delta\theta_{IJ} = \lambda_{x,I}(x_J - x_I) + \lambda_{y,I}(y_J - y_I) + \lambda_{z,I}(z_J - z_I)\,, \tag{5.22}$$

where x, y, z are the Cartesian coordinates of the cell centres (nodes in the case of the median-dual scheme). The Lagrange multipliers λ are computed for each cell (node) and follow from [2]

$$\begin{aligned}
\lambda_x &= \frac{R_x\,a_{11} + R_y\,a_{12} + R_z\,a_{13}}{d} \\
\lambda_y &= \frac{R_x\,a_{21} + R_y\,a_{22} + R_z\,a_{23}}{d} \\
\lambda_z &= \frac{R_x\,a_{31} + R_y\,a_{32} + R_z\,a_{33}}{d}
\end{aligned} \tag{5.23}$$

with the coefficients

$$\begin{aligned}
a_{11} &= I_{yy}I_{zz} - I_{yz}^2 \\
a_{12} &= I_{xz}I_{yz} - I_{xy}I_{zz} \\
a_{13} &= I_{xy}I_{yz} - I_{xz}I_{yy} \\
a_{21} &= I_{xz}I_{yz} - I_{xy}I_{zz} \\
a_{22} &= I_{xx}I_{zz} - I_{xz}^2 \\
a_{23} &= I_{xy}I_{xz} - I_{xx}I_{yz} \\
a_{31} &= I_{xy}I_{yz} - I_{xz}I_{yy} \\
a_{32} &= I_{xz}I_{xy} - I_{xx}I_{yz} \\
a_{33} &= I_{xx}I_{yy} - I_{xy}^2 \\
d &= I_{xx}I_{yy}I_{zz} - I_{xx}I_{yz}^2 - I_{yy}I_{xz}^2 - I_{zz}I_{xy}^2 + 2I_{xy}I_{xz}I_{yz}\,.
\end{aligned} \tag{5.24}$$

Written for a cell I, the first-order moments read

$$
\begin{aligned}
R_{x,I} &= \sum_{J=1}^{N_A} (x_J - x_I) \\
R_{y,I} &= \sum_{J=1}^{N_A} (y_J - y_I) \\
R_{z,I} &= \sum_{J=1}^{N_A} (z_J - z_I) \,.
\end{aligned}
\tag{5.25}
$$

Furthermore, the second-order moments are given by

$$
\begin{aligned}
I_{xx,I} &= \sum_{J=1}^{N_A} (x_J - x_I)^2 \\
I_{yy,I} &= \sum_{J=1}^{N_A} (y_J - y_I)^2 \\
I_{zz,I} &= \sum_{J=1}^{N_A} (z_J - z_I)^2 \\
I_{xy,I} &= \sum_{J=1}^{N_A} (x_J - x_I)(y_J - y_I) \\
I_{xz,I} &= \sum_{J=1}^{N_A} (x_J - x_I)(z_J - z_I) \\
I_{yz,I} &= \sum_{J=1}^{N_A} (y_J - y_I)(z_J - z_I) \,.
\end{aligned}
\tag{5.26}
$$

The geometrical weights (5.21) can lead to a non-positive approximation of the Laplacian and hence to a lost of stability on severely distorted grids. Therefore, clipping the weights to the range (0, 2) was suggested in [44]. However, this measure impairs the accuracy of the discretisation. See also the discussion in Ref. [12] for further details.

The fourth-order differences are evaluated as the Laplacian of the Laplacian, i.e., $L(U)$ is substituted for U in Eq. (5.20). Hence, the final form of the artificial dissipation term is for a cell I

$$
\begin{aligned}
\vec{D}_I = & \sum_{J=1}^{N_A} (\hat{\Lambda}_c)_{IJ}\, \epsilon_{IJ}^{(2)}\, \theta_{IJ} \left(\vec{W}_J - \vec{W}_I\right) \\
& - \sum_{J=1}^{N_A} (\hat{\Lambda}_c)_{IJ}\, \epsilon_{IJ}^{(4)}\, \theta_{IJ} \left[L(\vec{W}_J) - L(\vec{W}_I)\right] \,.
\end{aligned}
\tag{5.27}
$$

With the artificial dissipation term added, the system of equations in Eq. (5.2) becomes

$$\Omega_I \frac{d\vec{W}_I}{dt} = -\left[\sum_{m=1}^{N_F} (\vec{F}_c - \vec{F}_v)_m \Delta S_m\right] + \vec{D}_I + \vec{Q}_I \Omega_I \,, \tag{5.28}$$

where N_F denotes the number of the faces of the control volume (which may differ from the number of adjacent control volumes, e.g., if a quadrilateral face is divided into two triangles).

The second- and the fourth-order terms in Eq. (5.27) are scaled by the spectral radius of the convective flux Jacobian. According to Ref. [28], the spectral radius in cell I can be evaluated as

$$(\hat{\Lambda}_c)_I = \sum_{m=1}^{N_F} (|V_m| + c_m)\, \Delta S_m \,, \tag{5.29}$$

where V_m represents the contravariant velocity (2.22) and c_m the speed of sound, respectively. Both quantities are evaluated from flow variables averaged at the face. The spectral radius at the face of the control volume is obtained from

$$(\hat{\Lambda}_c)_{IJ} = \frac{1}{2}\left[(\hat{\Lambda}_c)_I + (\hat{\Lambda}_c)_J\right] . \tag{5.30}$$

A pressure-based sensor is used to switch off the fourth-order differences at shocks and the second-order differences in smooth portions of the flow field. Herewith, the coefficients $\epsilon_{IJ}^{(2)}$ and $\epsilon_{IJ}^{(4)}$ in Eq. (5.27) are defined as

$$\begin{aligned} \epsilon_{IJ}^{(2)} &= k^{(2)} \max(\Upsilon_I, \Upsilon_J) \\ \epsilon_{IJ}^{(4)} &= \max\left[0,\, (k^{(4)} - \epsilon_{IJ}^{(2)})\right] \end{aligned} \tag{5.31}$$

with the pressure sensor given by

$$\Upsilon_I = \frac{\left|\sum_{J=1}^{N_A} \theta_{IJ}\, (p_J - p_I)\right|}{\sum_{J=1}^{N_A} (p_J + p_I)} \tag{5.32}$$

Typical values of the parameters are $k^{(2)} = 1/2$ and $1/128 \leq k^{(4)} \leq 1/64$.

As we already discussed in Subsection 4.3.1, the accuracy of the above central scheme can be improved when we substitute a matrix [46] for the spectral radius $(\hat{\Lambda}_c)_{IJ}$ in Eq. (5.27). The implementation of this so-called *matrix dissipation* scheme on unstructured grids proceeds in the same way as on structured grids, with the scaling matrix defined as in Eq. (4.59). Application of the matrix dissipation scheme to 3-D mixed grids is discussed, e.g., in Ref. [47].

It is important to note that for elements other than triangles/tetrahedra, the popular explicit Runge-Kutta type of temporal discretisation experiences

severe stability problems when it is coupled to the central scheme [48]. The reason is the representation of the fourth-order differences by the Laplacian of the Laplacian. A remedy is to employ a difference of the left and the right state (cf. Section 4.3) for the approximation of the fourth-order differences [48], i.e.,

$$\vec{D}_I = \sum_{J=1}^{N_A} (\hat{\Lambda}_c)_{IJ}\, \epsilon_{IJ}^{(2)}\, \theta_{IJ} \left(\vec{W}_J - \vec{W}_I\right) - \sum_{J=1}^{N_A} 4\, (\hat{\Lambda}_c)_{IJ}\, \epsilon_{IJ}^{(4)} \left(\vec{W}_L - \vec{W}_R\right) . \quad (5.33)$$

This approach leads on quadrilateral/hexahedral grids to the same stencil as the corresponding structured scheme. The left and right state are computed using, e.g., the linear reconstruction described in Subsection 5.3.3.

5.3.2 Upwind Schemes

Upwind schemes seem to have gained, at least for the moment, much more popularity on unstructured grids than the above central scheme. In fact, the flux-difference splitting scheme of Roe [49] is the most widely employed approach on unstructured grids. It is the considerably more accurate resolution of boundary layers and the lower sensitivity to grid distortions in comparison to the central scheme, which explains the attractivity of Roe's scheme. However, the price to be paid for the improved performance is the higher computational effort, which becomes quite significant if a limiter has to be used to suppress oscillations of the solution (Subsection 5.3.5).

Any of the upwind schemes presented in Section 4.3 for structured grids are applicable to unstructured grids without modifications to the basic methodology. Only the computation of the left and right state (Eq. (5.18)), which is denoted as solution reconstruction, as well as the evaluation of the limiting function require new formulations. For this reason, only the solution reconstruction and the limiters are discussed here. For details on the various upwind methods, the reader is referred to Subsections 4.3.2–4.3.4. An example for the implementation of Roe's scheme on unstructured grids using the median-dual approach can be found in Ref. [33].

5.3.3 Solution Reconstruction

As we saw in Subsections 4.3.2–4.3.4, upwind schemes require flow states to be specified on the left and the right side of a control volume face. The same holds also for the modified artificial dissipation scheme in Eq. (5.33).

As a first approach, we can assume that the solution is constant inside each control volume. The left and right state are then simply the flow variables computed for the left and the right control volume. For example, in the case of the median-dual scheme (Fig. 5.9) we would set

$$\begin{aligned} U_L &= U_i \\ U_R &= U_j \end{aligned} \quad (5.34)$$

with U representing some scalar flow variable. This leads to a spatial discretisation which is only first-order accurate. For viscous flows, first-order accurate solution are too diffusive and lead to excessive growth of shear layers. Therefore, more accurate methods are required for the computation of these flows.

We can achieve second- and higher-order accuracy if we assume the solution to vary over the control volumes. For second-order accurate methods, which are the most commonly employed higher-order methods, the solution is assumed to vary in a linear fashion over the control volume. In order to compute the left and right state, a reconstruction of the assumed solution variation becomes necessary. In what follows, we shall discuss the most popular approaches for the reconstruction of linear and quadratic variations. The interested reader is referred to [37] for a comparison of various linear reconstruction techniques.

Reconstruction Based on MUSCL Approach

One possibility to achieve second-order accuracy consists of the extension of the MUSCL approach [50] to unstructured grids. When applied to the median-dual scheme, the method generates for each edge ij two "phantom" nodes i' and j' [51]-[55]. These phantom nodes are located at the endpoints of the line obtained by extending the edge ij by its length in both directions as sketched in Fig. 5.12. After the solution is interpolated from the surrounding elements (gray coloured in Fig. 5.12) to the phantom nodes, we can evaluate the left and right state using the MUSCL formulae Eq. (4.46). Hence,

$$\begin{aligned} U_R &= U_j - \frac{1}{4}\left[(1+\hat{\kappa})\Delta_- + (1-\hat{\kappa})\Delta_+\right] U_j \\ U_L &= U_i + \frac{1}{4}\left[(1+\hat{\kappa})\Delta_+ + (1-\hat{\kappa})\Delta_-\right] U_i \end{aligned} \tag{5.35}$$

with forward (Δ_+) and the backward (Δ_-) difference operators defined as

$$\begin{aligned} \Delta_+ U_i &= U_j - U_i \qquad & \Delta_- U_i &= U_i - U_{i'} \\ \Delta_+ U_j &= U_{j'} - U_j \qquad & \Delta_- U_j &= U_j - U_i \,. \end{aligned} \tag{5.36}$$

The MUSCL interpolation (5.35) has to be enhanced by a limiter function (according to Subsection 4.3.5) in the case of strong discontinuities. A disadvantage of this methodology is the necessity to store for each edge the elements which contain the phantom nodes. A further, conceptual, disadvantage is that no unique gradient is reconstructed for a control volume. Furthermore, difficulties can arise at boundaries, where one of the phantom points lies outside the physical domain.

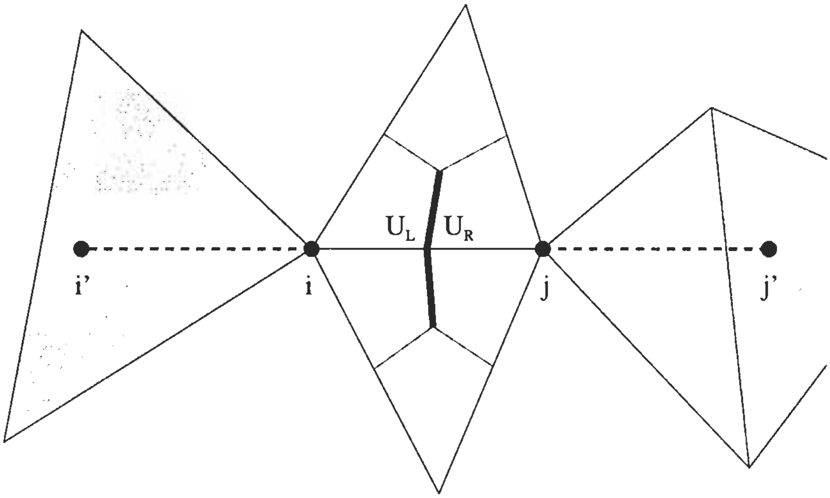


Figure 5.12: Evaluation of the left and right state based on interpolation from elements in the direction of an edge ij (median-dual scheme in 2D).

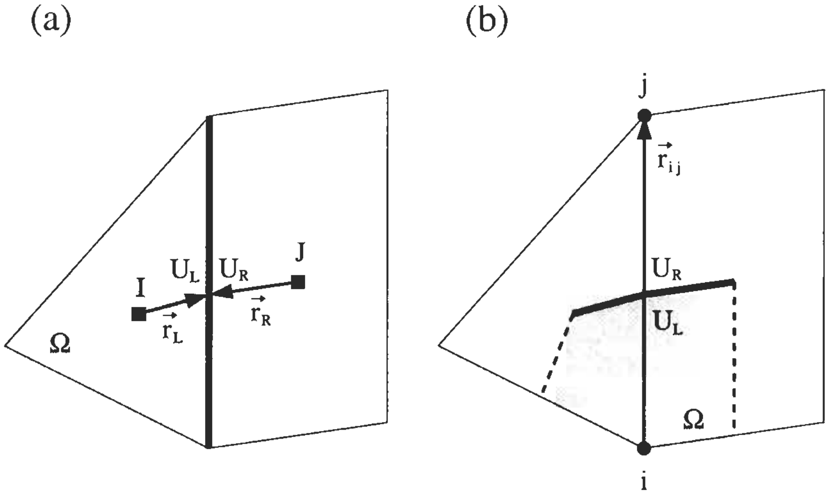


Figure 5.13: Linear reconstruction for the cell-centred (a) and the median-dual (b) scheme in 2D.

Piecewise Linear Reconstruction

Barth and Jespersen presented in [30] a reconstruction method, which is closely related to the finite element schemes. Here, it is assumed that the solution is piecewise linearly distributed over the control volume. Then, we can find the left and right state for a cell-centred scheme from the relations

$$\begin{aligned} U_L &= U_I + \Psi_I(\nabla U_I \cdot \vec{r}_L) \\ U_R &= U_J + \Psi_J(\nabla U_J \cdot \vec{r}_R)\,, \end{aligned} \tag{5.37}$$

where ∇U_I is the gradient of U ($= [\partial U/\partial x,\ \partial U/\partial y,\ \partial U/\partial z]^T$) at the cell centre I and Ψ denotes a limiter function (cf. Subsection 5.3.5), respectively. The vectors $\vec{r}_L$ and $\vec{r}_R$ point from the cell-centroid to the face-midpoint, as indicated in Fig. 5.13a.

The same approach applies to the median-dual scheme [30], i.e.,

$$\begin{aligned} U_L &= U_i + \frac{1}{2}\Psi_i(\nabla U_i \cdot \vec{r}_{ij}) \\ U_R &= U_j - \frac{1}{2}\Psi_j(\nabla U_j \cdot \vec{r}_{ij})\,. \end{aligned} \tag{5.38}$$

According to Fig. 5.9 or Fig. 5.13b,

$$\vec{r}_{ij} = \vec{r}_j - \vec{r}_i \tag{5.39}$$

represents the vector from node i to node j.

It can be seen easily that the method of Barth and Jespersen corresponds to a Taylor-series expansion around the neighbouring centres/nodes of the face, where only the linear term is retained. The linear reconstruction is formally second-order accurate on regular grids [37]. The scheme reconstructs a linear function exactly on any grid, provided the gradient ∇U is evaluated without an error. The linear reconstruction is likely the most popular one among the reconstruction methods.

The above scheme requires the computation of gradients at cell centres or at nodes, respectively. This can be accomplished either by the Green-Gauss or the least-squares approach, which are presented below in Subsection 5.3.4. Furthermore, the implementation of the limiter function on unstructured grids is described in detail in Subsection 5.3.5.

Linear Reconstruction Based on Nodal Weighting Procedure

It was demonstrated by Frink [26] that for the cell-centred scheme the linear reconstruction (5.37) does not require an explicit evaluation of the gradient on purely triangular or tetrahedral grids. The reason are two invariant geometric features of these elements. First, a line from a node through the cell-centroid will always intersect the midpoint of the opposing face. Second, the distance from the cell-centroid to the face-midpoint is one-fourth (one-third for a triangle) of

that from the face-midpoint to the opposing node. Thus, the gradient at the cell centre can be approximated by a simple finite difference [26]. For example, if we were to reconstruct the solution at the face-midpoint F_3 in Fig. 5.7a, the formulae (5.37) would become

$$U_{L/R} = U_C + \frac{\Psi_C}{4}\left[\frac{1}{3}(U_1 + U_2 + U_4) - U_3\right] \tag{5.40}$$

with U_C being the values at cell-centroid, U_1, U_2, etc. denoting the nodal values, and finally Ψ standing for a limiter.

Two different ways were devised by Frink in order to determine the nodal values. The first approach is based on inverse distance weighting. Here, the contribution to a node from the surrounding cells is inversely proportional to the distance from the node to the cell-centroid [26], [56], i.e.,

$$U_i = \Big(\sum_{J=1}^{N_A} \theta_{iJ} U_J\Big) \,/\, \Big(\sum_{J=1}^{N_A} \theta_{iJ}\Big)\,, \tag{5.41}$$

with the weights $\theta_{iJ} = 1/r_{iJ}$. The distance is computed from

$$r_{iJ} = \sqrt{(x_J - x_i)^2 + (y_J - y_i)^2 + (z_J - z_i)^2}\,. \tag{5.42}$$

The subscripts J and i refer to the cell-centroid and to the node, respectively. The above methodology leads to a reconstruction which is less than second-order accurate. However, Frink pointed out that no limiter is needed at least for inviscid flows [26], which reduces the computational effort significantly.

The second approach is based on work of Holmes et al. [44] and Rausch et al. [45] in 2D. It was later extended to 3D by Frink [2]. Here, the weights θ_{iJ} in Eq. (5.41) are defined such that the nodal values are computed exactly if the variation is linear. This leads to the same constraints as for the computation of the pseudo Laplacian (5.20). Consequently, the weights are also the same and follow from the Equations (5.21)-(5.26). The coordinates x_I, y_I, z_I of the cell-centroids are just replaced by the node coordinates x_i, y_i, z_i. The scheme is formally second-order accurate because the nodal values are computed exactly for a linear function. In order to assure positivity on distorted grids, the weights have to be restricted to the range (0, 2) [44]. Unfortunately, this reduces the accuracy of the reconstruction. Frink et al. [4] also reported recently some anomalous behaviour of the reconstruction for the Navier-Stokes equations.

Piecewise Quadratic Reconstruction

In order to achieve higher than second-order accuracy with a polynomial reconstruction, we have to keep further terms in the truncated Taylor-series expansion around the neighbouring cell-centres/nodes of the face. Based on the work of Barth and Frederickson [57], Barth developed the concept of k-exact reconstruction scheme [58], i.e., a reconstruction exact for a polynomial of degree k. The polynomial in Barth's method is defined in a way which guarantees the

conservation of the mean, or in other words, the average of the reconstruction polynomial is equal to the mean solution in the control volume. This property assures the conservation of mass, momentum, and energy during the reconstruction. The method was implemented for $k = 3$ in a median-dual scheme. The coefficients of the polynomial were computed using a least-squares approach. Similar ideas were followed for the cell-centred scheme by Mitchell and Walters [59], and by Mitchell [60]. However, these methods require a prohibitively high numerical effort and a complicated data structure which prevented widespread use.

Delanaye and Essers [61] and Delanaye [62] developed a particular form of quadratic reconstruction for the cell-centred scheme which is computationally more efficient than the method of Barth. The left and right state are approximated using Taylor series truncated after the quadratic term [61], [62]

$$
\begin{aligned}
U_L &= U_I + \Psi_{I,1}(\nabla U_I \cdot \vec{r}_L) + \frac{1}{2}\Psi_{I,2}(\vec{r}_L^{\,T}\, \bar{H}_I\, \vec{r}_L) \\
U_R &= U_J + \Psi_{J,1}(\nabla U_J \cdot \vec{r}_R) + \frac{1}{2}\Psi_{J,2}(\vec{r}_R^{\,T}\, \bar{H}_J\, \vec{r}_R)\,.
\end{aligned}
\tag{5.43}
$$

In the above Eq. (5.43), $\bar{H}_I$ denotes the Hessian matrix, i.e.,

$$
\bar{H}_I = \begin{bmatrix} \partial^2_{xx}U & \partial^2_{xy}U & \partial^2_{xz}U \\ \partial^2_{xy}U & \partial^2_{yy}U & \partial^2_{yz}U \\ \partial^2_{xz}U & \partial^2_{yz}U & \partial^2_{zz}U \end{bmatrix}_I ,
\tag{5.44}
$$

evaluated at the cell-centroid I. The variables $\Psi_{I,1}$ and $\Psi_{I,2}$ represent two different limiter functions for the linear and the quadratic term [61], respectively. The quadratic reconstruction method is third-order accurate on regular grids and at least second-order accurate on arbitrary grids due to cancellation of error terms [62]. Necessary conditions for achieving these properties are, however, that the gradient ∇U in Eq. (5.43) is evaluated at least with second-order and the Hessian with first-order accuracy. This is accomplished by combining Green-Gauss gradient evaluation with least-squares based approximation of the second derivatives [61], [62], which leads to a numerically efficient scheme. But the memory and time overheads are still quite significant in comparison to the linear reconstruction. The method utilises a fixed stencil composed of face and node neighbours. The stencil is shown in Fig. 5.14 together with the integration path employed for the Green-Gauss gradient computation. In order to determine all coefficients of the quadratic polynomial, at least six (ten in 3D) values must be provided by the stencil. To maintain the accuracy provided by the quadratic reconstruction, it is necessary to consider a linear variation of the solution over the face instead of a constant value. This implies that the solution must be reconstructed at two points – so-called *Gauss quadrature* points (cf. Fig. 5.14) – of a 2-D face (at three points of a triangular face) and that the fluxes have to be integrated in a piecewise manner over the face of the control volume [36].

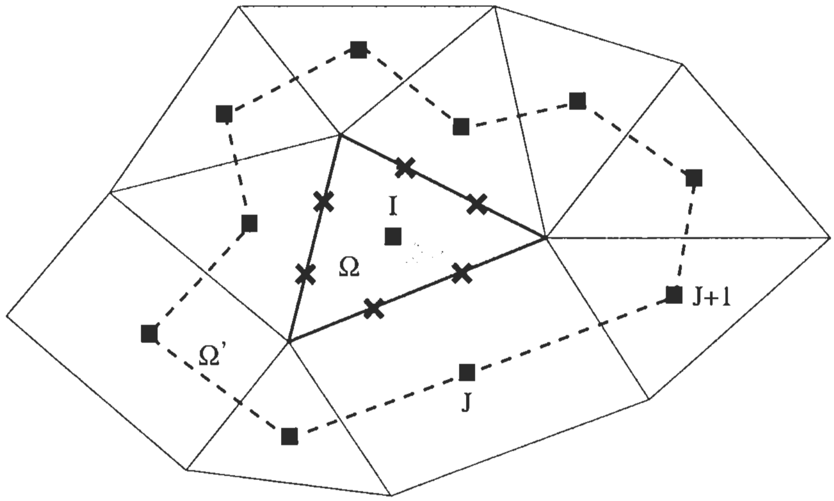


Figure 5.14: Stencil of the quadratic reconstruction method due to Delanaye [61], [62] in 2D (filled rectangles). Dashed line represents the integration path of the Green-Gauss gradient evaluation (control volume Ω'). Crosses denote the quadrature points for integration of the fluxes.

5.3.4 Evaluation of Gradients

An open point which remains from the discussion of the piecewise linear and the quadratic reconstruction is the determination of the gradient. Gradients of the velocity components and the temperature are also required for the evaluation of the viscous fluxes (Section 5.4). Two approaches will be presented in the following: the first is based on the Green-Gauss theorem and the second utilises the least-squares method.

Green-Gauss Approach

This method approximates the gradient of some scalar function U as the surface integral of the product of U with an outward-pointing unit normal vector over some control volume Ω', i.e.,

$$\nabla U \approx \frac{1}{\Omega'} \int_{\partial\Omega'} U \vec{n}\, dS\,. \tag{5.45}$$

Median-Dual Scheme

Barth and Jespersen [30] derived a particular discretisation of the Green-Gauss approach from the Galerkin finite element method. Later on, the discretisation was extended to 3D by Barth [63]. Barth and Jespersen applied Eq. (5.45) to the region formed by the union of the elements meeting at a node. They proved

that the approach can be formulated such that it becomes compatible with the edge-based data structure. However, this works only for the median-dual scheme on triangular/tetrahedral grids. The resulting formula reads

$$\nabla U_i \approx \frac{1}{\Omega} \sum_{j=1}^{N_F} \frac{1}{2}(U_i + U_j)\, \vec{n}_{ij} \Delta S_{ij} \,. \tag{5.46}$$

Here, Ω' in Eq. (5.45) equals to the volume of the median-dual control volume Ω. The summation extends over all N_F edges incident to node i. Furthermore, $\vec{n}_{ij}$ denotes the average unit normal vector according to Eq. (5.19), and ΔS_{ij} is the total face area, respectively. The same formula (5.46) is applicable in two or in three dimensions. It is important to mention that the summation has to be changed at boundaries in order to obtain a consistent approximation [64] (see also Section 8.9).

Cell-Centred Scheme

We can use the Green-Gauss method in the cell-centred scheme as well. Hence, the gradient at some cell-centroid I can be obtained from

$$\nabla U_I \approx \frac{1}{\Omega} \sum_{J=1}^{N_F} \frac{1}{2}(U_I + U_J)\, \vec{n}_{IJ} \Delta S_{IJ} \,, \tag{5.47}$$

where the summation extends over all faces of the cell with the volume Ω. In Eq. (5.47), $\vec{n}_{IJ}$ denotes the unit normal vector and ΔS_{IJ} the face area, respectively.

Mixed Grids

The main attractivity of the Green-Gauss gradient evaluation by Eq. (5.46) or (5.47) is its similarity to the computation of the fluxes (e.g., Eq. (5.15)). This means that no additional data structures are needed for the reconstruction of gradients. The main disadvantage is that the approximation in Eq. (5.46) or Eq. (5.47), respectively, fails on mixed grids. It was demonstrated in [48] that the gradient can become highly inaccurate, particularly where different element types meet. We can solve the problem in the case of the median-dual scheme when we keep the volume Ω' in Eq. (5.45) identical to the union of all cells incident to node i. Referring to the situation sketched in Fig. 5.8, the gradient results then in 2D from

$$\nabla U_i \approx \frac{1}{\Omega'} \sum_{j=1}^{N_O} \frac{1}{2}(U_j + U_{j+1})\, \vec{n}_j \Delta S_j \tag{5.48}$$

with $i = 0$, the number of outer faces $N_O = 6$, and $j{+}1 = 1$ for $j = 6$. Furthermore, $\vec{n}_j$ and ΔS_j stand for the unit normal vector and the area of the outer cell faces. In 3D, we can use

$$\nabla U_i \approx \frac{1}{\Omega'} \sum_{j=1}^{N_O} \frac{1}{3}(U_{j,1} + U_{j,2} + U_{j,3})\, \vec{n}_j \Delta S_j \,, \tag{5.49}$$

when we assume all faces are triangles – either naturally or by decomposition.

The same remedy can be also employed for the cell-centred scheme. The surface of the control volume Ω' is then defined by the centroids of the distant-one and distant-two neighbouring cells [61], [62], as it is rendered in Fig. 5.14. The gradient is computed correspondingly to Eq. (5.48) or (5.49) with cell instead of node indices.

The clear disadvantage of such a cell-based approach is the necessity of an additional data structure, which provides a link between the central node/centroid and the outer faces of Ω'. Thus, the approach is no longer grid-transparent (i.e., independent of cell information) and an efficient gather-scatter loop is no longer possible. This renders the least-squares technique, which is described below, more attractive on mixed grids.

Using the edge-/face-based implementation (5.46) or (5.47) on triangular or tetrahedral grids, and the cell-based methodology (5.48) or (5.49) on mixed grids, the Green-Gauss approach is at least first-order accurate [62]. It is also consistent, i.e., the gradient of a linear function is computed to roundoff error. First-order accuracy is sufficient for the linear reconstruction. Second-order accuracy on arbitrary grids, which is required for the quadratic reconstruction, can be achieved by subtracting an estimate of the truncation error from the first-order approximation of the gradient [62].

Least-Squares Approach

The evaluation of gradients by the least-squares approach was first introduced by Barth [63], [35]. In order to illustrate the method, let us consider the median-dual scheme. Herewith, the least-squares approach is based upon the use of a first-order Taylor series approximation for each edge which is incident to the central node i. The change of the solution along an edge ij can be computed from

$$(\nabla U_i) \cdot \vec{r}_{ij} = U_j - U_i \,, \tag{5.50}$$

where $\vec{r}_{ij}$ is given by Eq. (5.39) and represents the vector from node i to node j (see Fig. 5.9 or Fig. 5.13b). When we apply the relation (5.50) to all edges incident to node i, we obtain the following over-constrained system of linear equations

$$\begin{bmatrix} \Delta x_{i1} & \Delta y_{i1} & \Delta z_{i1} \\ \Delta x_{i2} & \Delta y_{i2} & \Delta z_{i2} \\ \vdots & \vdots & \vdots \\ \Delta x_{ij} & \Delta y_{ij} & \Delta z_{ij} \\ \vdots & \vdots & \vdots \\ \Delta x_{iN_A} & \Delta y_{iN_A} & \Delta z_{iN_A} \end{bmatrix} \begin{bmatrix} \partial_x U \\ \partial_y U \\ \partial_z U \end{bmatrix}_i = \begin{bmatrix} \theta_1 (U_1 - U_i) \\ \theta_2 (U_2 - U_i) \\ \vdots \\ \theta_j (U_j - U_i) \\ \vdots \\ \theta_{N_A} (U_{N_A} - U_i) \end{bmatrix} \tag{5.51}$$

with $\Delta(\cdot)_{ij} = (\cdot)_j - (\cdot)_i$ and $\partial_m(\cdot) = \partial(\cdot)/\partial m$. Further, N_A denotes the number of adjacent nodes j connected to i by an edge and θ_j stands for some weighting coefficient. The weights can depend on the geometry and/or on the solution

(see, e.g., [39]). However, in practice θ_j is usually set to unity. For convenience, we abbreviate the above system (5.51) as

$$\bar{A}\vec{x} = \vec{b}\,. \tag{5.52}$$

Solving Eq. (5.52) for the gradient vector $\vec{x}$ requires the inversion of the matrix $\bar{A}$. To prevent problems with ill-conditioning (particularly on stretched grids), Anderson and Bonhaus suggested to decompose $\bar{A}$ into the product of an orthogonal matrix $\bar{Q}$ and an upper triangular matrix $\bar{R}$ using the Gram-Schmidt process [65]. Their approach was recently extended to 3D in [48]. Hence, the solution to Eq. (5.52) immediately follows from

$$\vec{x} = \bar{R}^{-1}\bar{Q}^T\vec{b}\,. \tag{5.53}$$

Using a lower case letter with double subscripts to denote a matrix element, we may write the Gram-Schmidt orthogonalisation of the matrix $\bar{A} = [\vec{a}_1, \vec{a}_2, \vec{a}_3]$ as $\bar{Q} = [\vec{q}_1, \vec{q}_2, \vec{q}_3]$, where

$$\begin{aligned}
\vec{q}_1 &= \frac{1}{r_{11}}\,\vec{a}_1 \\
\vec{q}_2 &= \frac{1}{r_{22}}\left(\vec{a}_2 - \frac{r_{12}}{r_{11}}\,\vec{a}_1\right) \\
\vec{q}_3 &= \frac{1}{r_{33}}\left[\vec{a}_3 - \frac{r_{23}}{r_{22}}\,\vec{a}_2 - \left(\frac{r_{13}}{r_{11}} - \frac{r_{12}}{r_{11}}\frac{r_{23}}{r_{22}}\right)\vec{a}_1\right].
\end{aligned} \tag{5.54}$$

The entries in the upper triangular matrix $\bar{R}$ are obtained from

$$\begin{aligned}
r_{11} &= \sqrt{\sum_{j=1}^{N_A}(\Delta x_{ij})^2} \\
r_{12} &= \frac{1}{r_{11}}\sum_{j=1}^{N_A}\Delta x_{ij}\Delta y_{ij} \\
r_{22} &= \sqrt{\sum_{j=1}^{N_A}(\Delta y_{ij})^2 - r_{12}^2} \\
r_{13} &= \frac{1}{r_{11}}\sum_{j=1}^{N_A}\Delta x_{ij}\Delta z_{ij} \\
r_{23} &= \frac{1}{r_{22}}\left(\sum_{j=1}^{N_A}\Delta y_{ij}\Delta z_{ij} - \frac{r_{12}}{r_{11}}\sum_{j=1}^{N_A}\Delta x_{ij}\Delta z_{ij}\right) \\
r_{33} &= \sqrt{\sum_{j=1}^{N_A}(\Delta z_{ij})^2 - (r_{13}^2 + r_{23}^2)}\,.
\end{aligned} \tag{5.55}$$

Using Eqs. (5.53)-(5.55), the gradient at node i follows from the weighted sum of the edge differences

$$\nabla U_i \equiv \vec{x} = \sum_{j=1}^{N_A} \vec{w}_{ij}(U_j - U_i) \tag{5.56}$$

with the vector of weights $\vec{w}_{ij}$ defined as

$$\vec{w}_{ij} = \begin{bmatrix} \alpha_{ij,1} - \dfrac{r_{12}}{r_{11}}\,\alpha_{ij,2} + \beta\,\alpha_{ij,3} \\ \alpha_{ij,2} - \dfrac{r_{23}}{r_{22}}\,\alpha_{ij,3} \\ \alpha_{ij,3} \end{bmatrix} . \tag{5.57}$$

The terms in the above Equation (5.57) are given by

$$\begin{aligned} \alpha_{ij,1} &= \frac{\Delta x_{ij}}{r_{11}^2} \\ \alpha_{ij,2} &= \frac{1}{r_{22}^2}\left(\Delta y_{ij} - \frac{r_{12}}{r_{11}}\Delta x_{ij}\right) \\ \alpha_{ij,3} &= \frac{1}{r_{33}^2}\left(\Delta z_{ij} - \frac{r_{23}}{r_{22}}\Delta y_{ij} + \beta\Delta x_{ij}\right) , \end{aligned} \tag{5.58}$$

where

$$\beta = \frac{r_{12}r_{23} - r_{13}r_{22}}{r_{11}r_{22}} . \tag{5.59}$$

The formulation of the least-squares approach remains for a cell-centred scheme formally the same, only the nodes have to be substituted by cell-centroids. An example may be found in Ref. [16].

The least-squares approach is first-order accurate [62] on general grids. It is also consistent, i.e., the gradient of a linear function is computed to roundoff error, regardless of the type of the elements. Therefore, the method is particularly suited to mixed grids. The computational costs are comparable to those of the Green-Gauss approach, since only a vector-scalar multiplication (Eq. (5.56)) is needed within a single loop over faces/edges. However, we have to pre-compute and store the six entries (Eq. (5.55)) of the upper triangular matrix $\bar{R}$ at each node.

Experience shows that the least-squares approach requires some attention in the case of the median-dual scheme if prismatic or hexahedral cells are employed on a viscous wall. Consider Fig. 5.15 and assume that we want to compute the gradients of the velocity components at node i. It may become obvious that only the contribution from the edge ij is useful, since at other nodes connected to i by an edge $u = v = w = 0$. To increase the support of the stencil, we can insert so-called *virtual edges* [48], as they are rendered by dashed lines in Fig. 5.15. The virtual edges help to improve the accuracy and the robustness of the discretisation scheme considerably. It should be stressed that they are employed only for the gradient reconstruction but not for the flux computation.

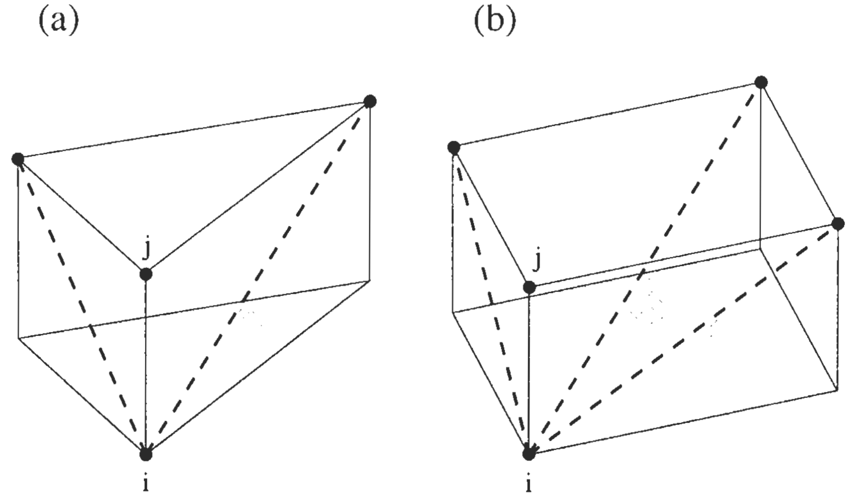


Figure 5.15: Virtual edges (dashed lines) used for the computation of gradients at node i [48]. Shown for prismatic (a) and hexahedral cell (b) on boundary (shaded).

5.3.5 Limiter Functions

Second- and higher-order upwind spatial discretisations require the use of so-called *limiters* or *limiter functions* in order to prevent the generation of oscillations and spurious solutions in regions of high gradients (e.g., at shocks). Hence, what we want to achieve is at least a *monotonicity preserving* scheme. This means that maxima in the flow field must be non-increasing, minima non-decreasing, and no new local extrema may be created during the time evolution. We discussed this point in Subsection 4.3.5 for the case of structured upwind schemes.

On unstructured grids, the purpose of a limiter is to reduce the gradient used to reconstruct the left and right state at the face of the control volume. The limiter function must be zero at strong discontinuities, in order to obtain a first-order upwind scheme which guarantees monotonicity. Setting the limiter to zero leads to the constant reconstruction of Eq. (5.34). Of course, the original unlimited reconstruction has to be retained in smooth flow regions, in order to keep the amount of numerical dissipation as low as possible. In the following, we shall describe two widely used limiter functions – namely the limiters of Barth and Jespersen [30] and of Venkatakrishnan [66], [67].

Limiter of Barth and Jespersen

The first implementation of a limiter function on unstructured grids was presented in [30]. In the case of the median-dual scheme, it is defined at node i as

$$\Psi_i = \min_j \begin{cases} \min\left(1, \dfrac{U_{\max} - U_i}{\Delta_2}\right) & \text{if } \Delta_2 > 0 \\[2ex] \min\left(1, \dfrac{U_{\min} - U_i}{\Delta_2}\right) & \text{if } \Delta_2 < 0 \\[2ex] 1 & \text{if } \Delta_2 = 0 \end{cases} \tag{5.60}$$

with the abbreviations

$$\begin{aligned} \Delta_2 &= \frac{1}{2}(\nabla U_i \cdot \vec{r}_{ij}) \\ U_{\max} &= \max(U_i,\ \max_j U_j) \\ U_{\min} &= \min(U_i,\ \min_j U_j)\,. \end{aligned} \tag{5.61}$$

In Equations (5.60) and (5.61), $\min_j$ or $\max_j$ means the minimum or maximum value of all direct neighbours j of node i (i.e., all nodes connected to i by an edge). Furthermore, the edge vector $\vec{r}_{ij}$, which is shown in Fig. 5.9 or in Fig. 5.13b, is defined according to Eq. (5.39). Finally, U_j denotes a scalar quantity at some neighbouring node j. Similar formulae to those above hold for the cell-centred scheme with cell instead of node indices and with

$$\Delta_2 = \nabla U_I \cdot \vec{r}_L\,, \tag{5.62}$$

where $\vec{r}_L$ denotes the vector from the cell-centroid to the midpoint of the corresponding cell face. In order to avoid division by a very small value of Δ_2 in Eq. (5.60), it is better to modify Δ_2 as $\text{Sign}(\Delta_2)(|\Delta_2|+\omega)$, where ω is approximately the machine accuracy [66].

Barth's limiter enforces a monotone solution. However, it is rather dissipative and it tends to smear discontinuities. A further problem presents the activation of the limiter due to numerical noise in smooth flow regions. This usually prevents the full convergence to steady state [66], [37]. Therefore, the limiter function due to Venkatakrishnan became more popular.

Venkatakrishnan's limiter

Venkatakrishnan's limiter [66], [67] is widely used because of its superior convergence properties. The limiter reduces the reconstructed gradient ∇U at the vertex i by the factor

$$\Psi_i = \min_j \begin{cases} \dfrac{1}{\Delta_2}\left[\dfrac{(\Delta_{1,\max}^2 + \epsilon^2)\Delta_2 + 2\Delta_2^2\Delta_{1,\max}}{\Delta_{1,\max}^2 + 2\Delta_2^2 + \Delta_{1,\max}\Delta_2 + \epsilon^2}\right] & \text{if } \Delta_2 > 0 \\[2ex] \dfrac{1}{\Delta_2}\left[\dfrac{(\Delta_{1,\min}^2 + \epsilon^2)\Delta_2 + 2\Delta_2^2\Delta_{1,\min}}{\Delta_{1,\min}^2 + 2\Delta_2^2 + \Delta_{1,\min}\Delta_2 + \epsilon^2}\right] & \text{if } \Delta_2 < 0 \\[2ex] 1 & \text{if } \Delta_2 = 0 \end{cases} \tag{5.63}$$

where

$$\begin{aligned} \Delta_{1,\max} &= U_{\max} - U_i \\ \Delta_{1,\min} &= U_{\min} - U_i \,. \end{aligned} \tag{5.64}$$

In the above Eq. (5.64), $U_{\max}$ and $U_{\min}$ stand for the minimum/maximum values of all surrounding nodes j and including the node i itself. Definitions of $U_{\max}$, $U_{\min}$ and Δ_2 are given in Eq. (5.61). The parameter ϵ^2 is intended to control the amount of limiting. Setting ϵ^2 to zero results in full limiting, but this may stall the convergence. Contrary to that, if ϵ^2 is set to a large value, the limiter function will return a value of about unity. Hence, there will be no limiting at all and wiggles could occur in the solution. In practice, it was found that ϵ^2 should be proportional to a local length scale, i.e.,

$$\epsilon^2 = (K\,\Delta h)^3\,, \tag{5.65}$$

where K is a constant of $\mathcal{O}(1)$ and Δh is for example the cube-root of the volume (square-root of the area in 2D) of the control volume. It is important to notice that the limiter function (5.63) must be defined with non-dimensional quantities. The influence of the coefficient K in Eq. (5.65) on the resolution of a shock is demonstrated in Fig. 5.16. It can be seen that the fully limited ($K = 0$) and the solution for $K = 5$ are identical. However, the explicit time-stepping scheme converged only about three orders of magnitude for $K = 0$, whereas for $K = 5$ it converged to machine zero (Fig. 5.17). Figure 5.16 also shows that the solution becomes gradually unlimited with increasing values of K. This manifests itself as an increasing overshoot at the shock.

The computational effort for the evaluation of one of the above limiter functions is relatively high. Two loops over edges (faces in the case of the cell-centred scheme) and one loop over nodes (cells) are necessary in order to compute $U_{\max}$, $U_{\min}$ as well as the limiter Ψ itself. Furthermore, $U_{\max}$, $U_{\min}$ and Ψ have to be stored node-(cell-)wise separately for each flow variable.

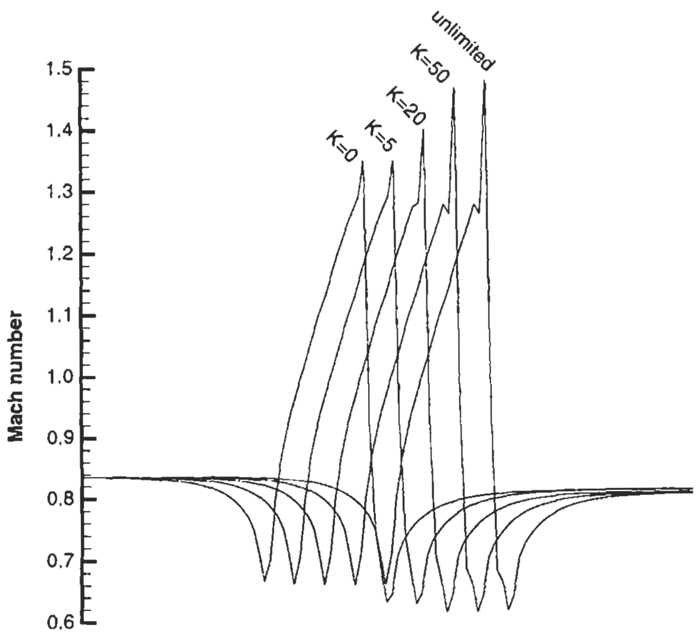


Figure 5.16: Effect of the constant K in Venkatakrishnan's limiter, given by Eq. (5.63), on the solution – inviscid flow past a circular arc.

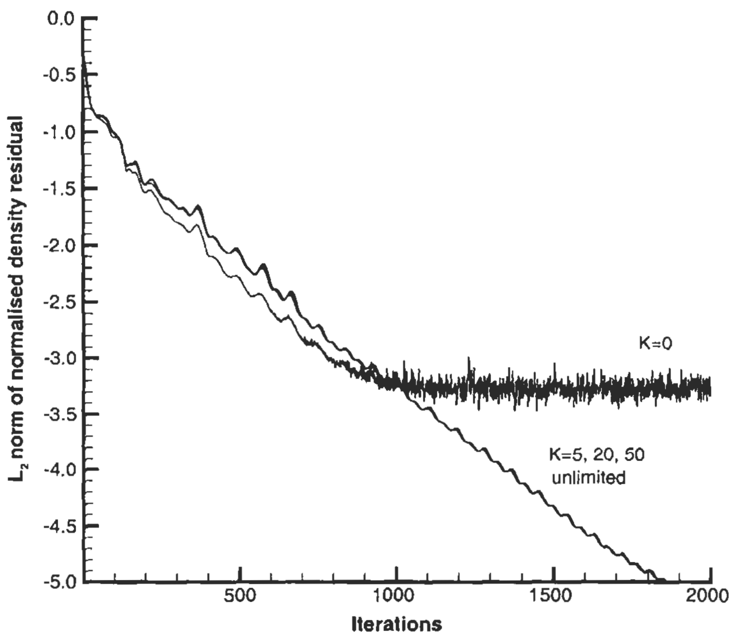


Figure 5.17: Effect of the constant K in Venkatakrishnan's limiter on the convergence – inviscid flow past a circular arc.

5.4 Discretisation of Viscous Fluxes

In order to evaluate the diffusive fluxes $\vec{F}_v$ in Eq. (5.2), flow quantities and their first derivatives have to be known at the faces of the control volumes. The control volume for the viscous fluxes is conveniently chosen to be the same as for the convective fluxes in order to obtain a consistent spatial discretisation and to simplify the data structure. Because of the elliptic nature of the viscous fluxes, values of the velocity components (u, v, w), the dynamic viscosity μ, and of the heat conduction coefficient k, which are required for the computation of the viscous terms (2.23), (2.24) and of the stresses (2.15), are simply averaged at a face. Thus, in the case of the cell-centred scheme (Fig. 5.13a), the values at the face IJ of the control volume result from

$$U_{IJ} = \frac{1}{2}(U_I + U_J)\,, \tag{5.66}$$

where U is any of the above flow variables. A similar expression holds in the case of the median-dual scheme for the face ij – see Fig. 5.13b.

The remaining task is the evaluation of the first derivatives (gradients) of the velocity components in Eq. (2.15) and of temperature in Eq. (2.24). This can be accomplished in one of two ways, i.e., by using

- element-based gradients, or
- average of gradients.

In the following, we shall learn more about both approaches.

5.4.1 Element-Based Gradients

A common feature of this type of gradient computation is the necessity to store either information about the grid elements or some coefficients related to the geometry of the elements. Hence, we have to extend the data structure beyond the face-/edge-based formulation presented earlier for the convective fluxes. Below, we discuss three well-established methods for the cell-centred and the median-dual discretisation.

Face-Centred Control Volume

One possible way of evaluating the gradients at a face of the control volume is to define an auxiliary control volume centred at the face and to employ the Green-Gauss theorem. We already discussed this approach in Section 4.4 in the framework of the structured finite volume discretisation. For example in the case of the median-dual scheme, we can compute the gradient at the edge-midpoint as the volume average of gradients for all elements which share the edge [68]. The element-based gradients are evaluated according to Eq. (5.45) by looping over all grid cells and accumulating the gradients at edges. The values of U at the cell-faces are obtained by averaging the nodal values, in a manner similar to Eq. (5.49). This approach is quite costly in terms of memory and number of operations. However, it can be implemented for any mix of grid elements.

Approximate Galerkin Finite Element Approach

Another methodology, which is applicable to the median-dual scheme, was derived from the Galerkin finite element method [31]. Basically speaking, the approach transforms the integration of gradients over the surface of the control volume into an evaluation of Hessian matrix (second derivatives) at the central node. The viscous terms then follow the differential form of the Navier-Stokes equations in Cartesian coordinates (Eq. (A.4) with $\xi = x$, $\eta = y$, $\zeta = z$ and $J^{-1} = 1$), which contains terms such as

$$\partial_x(\mu\,\partial_x u)\,,\ \text{etc.}$$

with $\partial_m(\cdot) = \partial(\cdot)/\partial m$. Hence, no further integration of the viscous fluxes over the faces of the control volume is required.

The original scheme was formulated for purely triangular/tetrahedral grids. It employs a union of all elements that contain the particular node. In order to simplify the implementation, the dynamic viscosity coefficient is averaged from the nodal values, which is a difference to the Galerkin method. Then, the second derivatives can be evaluated at node i as follows [69]

$$\begin{bmatrix} \partial_x(\mu\,\partial_x U) & \partial_y(\mu\,\partial_x U) & \partial_z(\mu\,\partial_x U) \\ \partial_x(\mu\,\partial_y U) & \partial_y(\mu\,\partial_y U) & \partial_z(\mu\,\partial_y U) \\ \partial_x(\mu\,\partial_z U) & \partial_y(\mu\,\partial_z U) & \partial_z(\mu\,\partial_z U) \end{bmatrix}_i = \frac{1}{\Omega'}\sum_{j=1}^{N_A}\left\{ \begin{bmatrix} \alpha_{xx} & \alpha_{xy} & \alpha_{xz} \\ \alpha_{yx} & \alpha_{yy} & \alpha_{yz} \\ \alpha_{zx} & \alpha_{zy} & \alpha_{zz} \end{bmatrix}_{ij} \frac{\mu_i + \mu_j}{2}\,(U_i - U_j) \right\}. \tag{5.67}$$

The volume Ω' contains all tetrahedra which share the node i. The coefficient matrix $\bar{\alpha}$ in Eq. (5.67) is symmetric about the diagonal [69], i.e., $\alpha_{xy} = \alpha_{yx}$, $\alpha_{xz} = \alpha_{zx}$, and $\alpha_{yz} = \alpha_{zy}$, respectively. Thus, it is necessary to store only six coefficients for each edge. The coefficients are given by [69]

$$\alpha_{nk} = \sum_e \frac{(S_e^j)_n\,(S_e^i)_k}{\Omega_e}\,, \tag{5.68}$$

where n, k denote the x, y, z subscripts, and $(S_e^i)_k$, $(S_e^j)_n$ represent components of the outer face vectors $\vec{S}_e^i$, $\vec{S}_e^j$ displayed in Fig. 5.18. The summation is carried out over all tetrahedra (with particular volumes Ω_e) which share the edge ij. If the grid is stationary, the coefficients can be computed in a pre-processing step. A desirable feature is that the same edge-based data structure can be employed as for the convective fluxes.

The disadvantage of this approach is that the full viscous terms are retained only on triangular or tetrahedral grids. For other elements like prisms or hexahedra, this technique simplifies to a TSL-like approximation of the Navier-Stokes equations in all three coordinate directions [8]. An extension to non-simplicial

elements which conserves the full viscous terms was presented in [70]. However, the efficient edge-based data structure can no longer be used since the stencil involves also nodes which are not directly connected by an edge.

Average of Nodal Values

This scheme is intended for the cell-centred type of control volume and purely tetrahedral grids. It employs a modified version [4] of the stencil introduced in [60] to evaluate the gradients at the cell faces. The approach is based on an average of the values at the three nodes which define the cell face, combined with the quantities at the cell centroids. The first derivatives at a cell face result from the solution of the linear equation system [4]

$$\begin{bmatrix} x_J - x_I & y_J - y_I & z_J - z_I \\ \frac{1}{2}(x_2 + x_3) - x_1 & \frac{1}{2}(y_2 + y_3) - y_1 & \frac{1}{2}(z_2 + z_3) - z_1 \\ \frac{1}{2}(x_1 + x_3) - x_2 & \frac{1}{2}(y_1 + y_3) - y_2 & \frac{1}{2}(z_1 + z_3) - z_2 \end{bmatrix} \begin{bmatrix} \partial_x U \\ \partial_y U \\ \partial_z U \end{bmatrix} = \begin{bmatrix} U_J - U_I \\ \frac{1}{2}(U_2 + U_3) - U_1 \\ \frac{1}{2}(U_1 + U_3) - U_2 \end{bmatrix}, \tag{5.69}$$

Referring to Fig. 5.19, the subscripts I, J denote the cell-centroids, and the subscripts $1, 2, 3$ stand for the nodes P_1, P_2 and P_3, respectively. Flow variables at the nodes can be determined either from inverse-distance weighting (Eq. (5.41)) or by pseudo-Laplacian weighting [2] (similar to Eq. (5.20)).

5.4.2 Average of Gradients

If we already computed the gradients inside each control volume (e.g., using the piecewise linear reconstruction, Eq. (5.37) or (5.38)), it would be tempting to evaluate the gradient at the face-midpoint by the simple average [71]

$$\overline{\nabla U}_{IJ} = \frac{1}{2}\left[\nabla U_I + \nabla U_J\right]. \tag{5.70}$$

This approach is particularly attractive, because it requires only the basic face-/edge-based data structure and no additional storage. However, as pointed out in, e.g., [69], it leads to a wide stencil with an unfavourable weight distribution [48]. Furthermore, it was demonstrated in [48] that the stencil allows the decoupling of the solution on quadrilateral or hexahedral grids.

The properties of the method can be improved and particularly the decoupling can be prevented by using the directional derivative along the connection

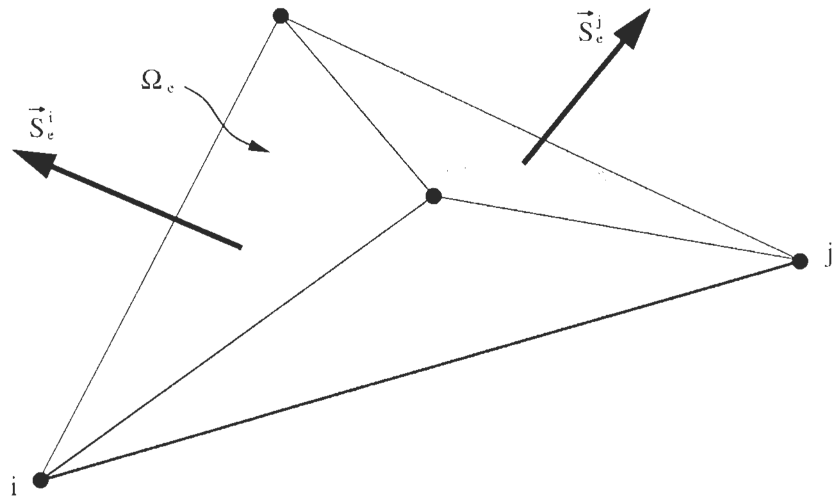


Figure 5.18: Viscous terms at node i: tetrahedron with volume Ω_e and the triangular faces involved in the computation of coefficients associated with the edge ij [69].

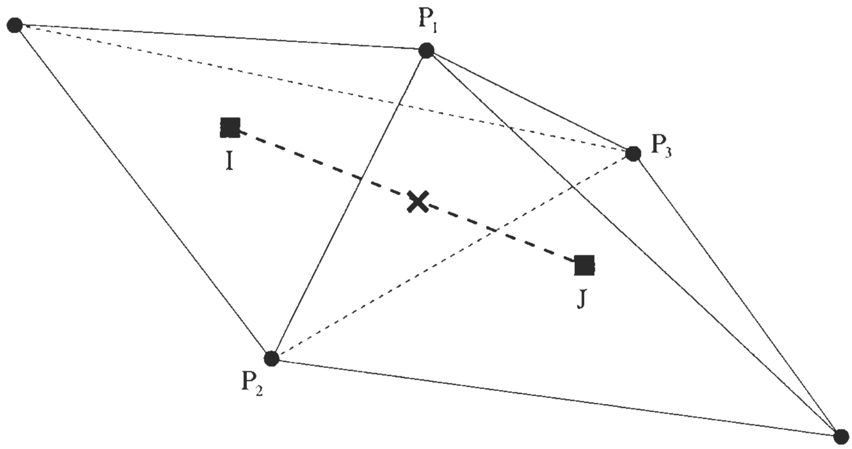


Figure 5.19: Cell-centred scheme: stencil for the computation of gradients on tetrahedral grids [4]. Cross denotes location where the gradients are evaluated in order to compute the viscous fluxes (face-midpoint $P_1P_2P_3$).

between the cell-centroids (in the case of the cell-centred scheme), i.e.,

$$\left(\frac{\partial U}{\partial \ell}\right)_{IJ} \approx \frac{U_J - U_I}{\ell_{IJ}}, \tag{5.71}$$

where ℓ_{IJ} represents the distance between the both cell-centroids I and J (dashed line in Fig. 5.19). A similar expression holds also for the median-dual scheme with $\vec{r}_{ij}$ being defined in Eq. (5.39). With the definition of the unit vector $\vec{t}_{IJ}$ along the line connecting I and J,

$$\vec{t}_{IJ} = \frac{\vec{r}_{IJ}}{\ell_{IJ}}, \tag{5.72}$$

the modified average may be written as [72], [73]

$$\nabla U_{IJ} = \overline{\nabla U}_{IJ} - \left[\overline{\nabla U}_{IJ} \cdot \vec{t}_{IJ} - \left(\frac{\partial U}{\partial \ell}\right)_{IJ}\right] \vec{t}_{IJ} \tag{5.73}$$

where $\overline{\nabla U}_{IJ}$ is given by Eq. (5.70). The modification leads to strongly coupled stencils on tetrahedral as well as on prismatic or hexahedral grids [48]. The modified approach is also still compatible with the face-/edge-based data structure and requires no additional storage. It is therefore more attractive than the element-based methodology, provided the gradients inside control volumes are utilised for the convective fluxes anyway.

Bibliography

[1] Hirsch, C.: *Numerical Computation of Internal and External Flows.* Vols. 1 and 2, John Wiley and Sons, 1988.

[2] Frink, N.T.: *Recent Progress Toward a Three-Dimensional Navier-Stokes Solver.* AIAA Paper 94-0061, 1994.

[3] Hassan, O.; Probert, E. J.; Weatherill, N. P.; Marchant, M. J.; Morgan, K.; Marcum, D. L.: *The Numerical Simulation of Viscous Transonic Flows Using Unstructured Grids.* AIAA Paper 94-2346, 1994.

[4] Frink, N.T.; Pirzadeh, S.Z.: *Tetrahedral Finite-Volume Solutions to the Navier-Stokes Equations on Complex Configurations.* 10th Int. Conf. on Finite Elements in Fluids, Tucson, USA, 1998.

[5] Luo, H.; Baum, J.D.; Löhner, R.: *Computation of Compressible Flows Using a Two-Equation Turbulence Model on Unstructured Grids.* AIAA Paper 97-0430, 1997.

[6] Wang, Q.; Massey, S.J.; Abdol-Hamid, K.S.; Frink, N.T.: *Solving Navier-Stokes Equations with Advanced Turbulence Models on Three-Dimensional Unstructured Grids.* AIAA Paper 99-0156, 1999.

[7] Luo, H.; Baum, J.D.; Löhner, R.: *A Fast, Matrix-Free Implicit Method for Compressible Flows on Unstructured Grids.* AIAA Paper 99-0936, 1999.

[8] Mavriplis, D.J.; Venkatakrishnan, V.: *A Unified Multigrid Solver for the Navier-Stokes Equations on Mixed Element Meshes.* ICASE Report No. 95-53, 1995.

[9] Khawaja A., Kallinderis Y. and Parthasarathy V., *Implementation of Adaptive Hybrid Grids for 3-D Turbulent Flows*, AIAA Paper 96-0026, 1996.

[10] Coirier W.J. and Jorgenson P.C.E., *A Mixed-Volume Approach for the Euler and Navier-Stokes Equations*, AIAA Paper 96-0762, 1996.

[11] Mavriplis, D.J.: *Adaptive Meshing Technique for Viscous Flow Calculations on Mixed-Element Unstructured Meshes.* AIAA Paper 97-0857, 1997.

[12] Haselbacher, A.C.; McGuirk, J.J.; Page, G.J.: *Finite Volume Discretisation Aspects for Viscous Flows on Mixed Unstructured Grids.* AIAA Paper 97-1946, 1997; also AIAA Journal, 37 (1999), pp. 177-184.

[13] Kano, S.; Nakahashi, K.: *Navier-Stokes Computations of HSCT Off-Design Aerodynamics Using Unstructured Hybrid Meshes.* AIAA Paper 98-0232, 1998.

[14] Sharov, D.; Nakahashi, K.: *Hybrid Prismatic/Tetrahedral Grid Generation for Viscous Flow Applications.* AIAA Journal, 36 (1998), pp. 157-162.

[15] Blazek, J.; Irmisch, S.; Haselbacher, A.: *Unstructured Mixed-Grid Navier-Stokes Solver for Turbomachinery Applications.* AIAA Paper 99-0664, 1999.

[16] Strang, W.Z.; Tomaro, R.F.; Grismer, M.J.: *The Defining Methods of* $Cobalt_{60}$*: A Parallel, Implicit, Unstructured Euler/Navier-Stokes Flow Solver.* AIAA Paper 99-0786, 1999.

[17] Nakahashi, K.: *FDM-FEM Zonal Approach for Computations of Compressible Viscous Flows.* Lecture Notes in Physics, 264 (1986), Springer Verlag, pp. 494-498.

[18] Nakahashi, K.: *A Finite-Element Method on Prismatic Elements for the Three-Dimensional Navier-Stokes Equations.* Lecture Notes in Physics, 323 (1989), Springer Verlag, pp. 434-438.

[19] Soetrisno, M.; Imlay, S.T.; Roberts, D.W.; Taflin, D.E.: *Computations of Viscous Flows for Multi-Element Wings Using Hybrid Structured-Unstructured Grids.* AIAA Paper 97-0623, 1997.

[20] Kallinderis, Y.; Khawaja, A.; McMorris, H.: *Hybrid Prismatic/Tetrahedral Grid Generation for Viscous Flows Around Complex Geometries.* AIAA Journal, 34 (1996), pp. 291-298.

[21] Kallinderis, Y.; Khawaja, A.; McMorris, H.; Irmisch, S.; Walker, D.: *Hybrid Prismatic/Tetrahedral Grids for Turbomachinery Applications.* Proc. 6th Int. Meshing Roundtable, Park City, Utah, USA, 1997, pp. 21-31.

[22] Baker, T.J.: *Discretization of the Navier-Stokes Equations and Mesh Induced Errors.* Proc. 5th Int. Conf. on Numerical Grid Generation in CFD, Mississipi State University, Mississipi, April 1996.

[23] Baker, T.J.: *Irregular Meshes and the Propagation of Solution Errors.* Proc. 15th Int. Conf. on Numerical Methods in Fluid Dynamics, Monterey, CA, June 1996.

[24] Mohanraj, R.; Neumeier, Y.; Zinn, B.T.: *Characteristic-Based Treatment of Source Terms in Euler Equations for Roe Scheme.* AIAA Journal, 37 (1999), pp. 417-424.

[25] Jameson, A.; Mavriplis, D.: *Finite Volume Solution of the Two-Dimensional Euler Equations on a Regular Triangular Mesh.* AIAA Journal, 24 (1986), pp. 611-618.

[26] Frink, N.T.; Parikh, P.; Pirzadeh, S.: *A Fast Upwind Solver for the Euler Equations on Three-Dimensional Unstructured Meshes.* AIAA Paper 91-0102, 1991.

[27] Mavriplis, D.J.: *Multigrid Solution of the Two-Dimensional Euler Equations on Unstructured Triangular Meshes.* AIAA Journal, 26 (1988), pp. 824-831.

[28] Mavriplis, D.J., Jameson, A.; Martinelli, L.: *Multigrid Solution of the Navier-Stokes Equations on Triangular Meshes.* ICASE Report, No. 89-11, 1989.

[29] Whitaker, D.L.; Grossman, B.; Löhner, R.: *Two-Dimensional Euler Computations on a Triangular Mesh Using an Upwind, Finite-Volume Scheme.* AIAA Paper 89-0470, 1989.

[30] Barth, T.J.; Jespersen, D.C.: *The Design and Application of Upwind Schemes on Unstructured Meshes.* AIAA Paper 89-0366, 1989.

[31] Barth, T.J.: *Numerical Aspects of Computing High-Reynolds Number Flows on Unstructured Meshes.* AIAA Paper 91-0721, 1991.

[32] Hwang, C.J.; Wu, S.J.: *Adaptive Finite Volume Upwind Approach on Mixed Quadrilateral-Triangular Meshes.* AIAA Journal, 31 (1993), pp. 61-67.

[33] Whitaker, D.L.: *Three-Dimensional Unstructured Grid Euler Computations Using a Fully-Implicit, Upwind Method.* AIAA Paper 93-3337, 1993.

[34] Bruner, C.W.S.: *Geometric Properties of Arbitrary Polyhedra in Terms of Face Geometry.* AIAA Journal, 33 (1995), p. 1350.

[35] Barth, T.J.: *Aspects of Unstructured Grids and Finite-Volume Solvers for the Euler and Navier-Stokes Equations.* AGARD R-787, Special Course on Unstructured Grid Methods for Advection Dominated Flows, Brussels, Belgium, 18-22 May, 1992, pp. 6.1-6.61.

[36] Essers, J.A.; Delanaye, M.; Rogiest, P.: *An Upwind-Biased Finite-Volume Technique for Solving Compressible Navier-Stokes Equations on Irregular Meshes. Applications to Supersonic Blunt-Body Flows and Shock-Boundary Layer Interactions.* AIAA Paper 93-3377, 1993.

[37] Aftosmis, M.; Gaitonde, D.; Tavares, T.S.: *Behavior of Linear Reconstruction Techniques on Unstructured Meshes.* AIAA Journal, 33 (1995), pp. 2038-2049.

[38] Xia, X.; Nicolaides, R.: *Covolume Techniques for Anisotropic Media.* Numer. Math., 61 (1992), pp. 215-234.

[39] Barth, T.J.; Linton, S.W.: *An Unstructured Mesh Newton Solver for Compressible Fluid Flow and Its Parallel Implementation.* AIAA Paper 95-0221, 1995.

[40] Venkatakrishnan, V.: *Implicit Schemes and Parallel Computing in Unstructured Grid CFD.* ICASE Report No. 95-28, 1995.

[41] Venkatakrishnan, V.; Mavriplis, D.J.: *Implicit Method for the Computation of Unsteady Flows on Unstructured Grids.* J. Computational Physics, 127 (1996), pp. 380-397.

[42] Jameson, A.; Schmidt, W.; Turkel, E.: *Numerical Solutions of the Euler Equations by Finite Volume Methods Using Runge-Kutta Time-Stepping Schemes.* AIAA Paper 81-1259, 1981.

[43] Jameson, A.; Baker, T.J.; Weatherill, N.P.: *Calculation of Inviscid Transonic Flow over a Complete Aircraft.* AIAA Paper 86-0103, 1986.

[44] Holmes, D.G. and Connell, S.D.: *Solution of the 2D Navier-Stokes Equations on Unstructured Adaptive Meshes.* AIAA Paper, No. 89-1932, 1989.

[45] Rausch, R.D.; Batina, J.T.; Yang, H.T.Y.: *Spatial Adaption Procedures on Unstructured Meshes for Accurate Unsteady Aerodynamic Flow Computations.* AIAA Paper 91-1106, 1991.

[46] Swanson, R.C.; Turkel, E.: *On Central Difference and Upwind Schemes.* J. Computational Physics, 101 (1992), pp. 292-306.

[47] Mavriplis, D.J.: *Directional Agglomeration Multigrid Techniques for High Reynolds Number Viscous Flow Solvers.* AIAA Paper 98-0612, 1998.

[48] Haselbacher, A.; Blazek, J.: *On the Accurate and Efficient Discretisation of the Navier-Stokes Equations on Mixed Grids.* AIAA Paper 99-3363, 1999; also AIAA Journal, 38 (2000), pp. 2094-2102.

[49] Roe, P.L.: *Approximate Riemann Solvers, Parameter Vectors, and Difference Schemes.* J. Computational Physics, 43 (1981), pp. 357-372.

[50] Van Leer, B.: *Towards the Ultimate Conservative Difference Scheme V. A second Order Sequel to Godunov's method.* J. Computational Physics, 32 (1979), pp. 101-136.

[51] Desideri, J.A.; Dervieux, A.: *Compressible Flow Solvers using Unstructured Grids.* VKI Lecture Series 1988-05, 1988, pp. 1-115.

[52] Fezoui, L.; Stoufflet, B.: *A Class of Implicit Upwind Schemes for Euler Simulations with Unstructured Meshes.* J. Computational Physics, 84 (1989), pp. 174-206.

[53] Whitaker D.L.; Slack, D.C.; Walters, R.W.: *Solution Algorithms for the Two-Dimensional Euler Equations on Unstructured Meshes.* AIAA Paper 90-0697, 1990.

[54] Arminjon, P.; Dervieux, A.: *Construction of TVD-like Artificial Viscosities on Two-Dimensional Arbitrary FEM Grids.* J. Computational Physics, 106 (1993), pp. 176-198.

[55] Jameson, A.: *Positive Schemes and Shock Modelling for Compressible Flows.* Int. J. Numerical Methods in Fluids, 20 (1995), pp. 743-776.

[56] Frink, N.T.: *Upwind Scheme for Solving the Euler Equations on Unstructured Tetrahedral Meshes.* AIAA Journal, 30 (1992), pp. 70-77.

[57] Barth, T.J.; Frederickson, P.O.: *Higher Order Solution of the Euler Equations on Unstructured Grids Using Quadratic Reconstruction.* AIAA Paper 90-0013, 1990.

[58] Barth, T.J.: *Recent Developments in High Order k-Exact Reconstruction on Unstructured Meshes.* AIAA Paper 93-0668, 1993.

[59] Mitchell, C.R.; Walters, R.W.: *K-Exact Reconstruction for the Navier-Stokes Equations on Arbitrary Grids.* AIAA Paper 93-0536, 1993.

[60] Mitchell, C.R.: *Improved Reconstruction Schemes for the Navier-Stokes Equations on Unstructured Meshes.* AIAA Paper 94-0642, 1994.

[61] Delanaye, M.; Essers, J.A.: *Finite Volume Scheme with Quadratic Reconstruction on Unstructured Adaptive Meshes Applied to Turbomachinery Flows.* ASME IGTI Gas Turbine Conference, Houston, USA, 1995.

[62] Delanaye, M.: *Polynomial Reconstruction Finite Volume Schemes for the Compressible Euler and Navier-Stokes Equations on Unstructured Adaptive Grids.* PhD Thesis, The University of Liège, Belgium, 1996.

[63] Barth, T.J.: *A 3-D Upwind Euler Solver for Unstructured Meshes.* AIAA Paper 91-1548, 1991.

[64] Luo, H.; Baum, J.D.; Löhner, R.: *An Improved Finite Volume Scheme for Compressible Flows on Unstructured Grids.* AIAA Paper 95-0348, 1995.

[65] Anderson, W.K.; Bonhaus, D.L.: *An Implicit Upwind Algorithm for Computing Turbulent Flows on Unstructured Grids.* Computers & Fluids, 23 (1994), pp. 1-21.

[66] Venkatakrishnan, V.: *On the Accuracy of Limiters and Convergence to Steady State Solutions.* AIAA Paper 93-0880, 1993.

[67] Venkatakrishnan, V.: *Convergence to Steady-State Solutions of the Euler Equations on Unstructured Grids with Limiters.* J. Computational Physics, 118 (1995), pp. 120-130.

[68] Sharov, D.; Nakahashi, K.: *Low Speed Preconditioning and LU-SGS Scheme for 3-D Viscous Flow Computations on Unstructured Grids.* AIAA Paper 98-0614, 1998.

[69] Mavriplis, D.J.: *Three-Dimensional Multigrid Reynolds-Averaged Navier-Stokes Solver for Unstructured Meshes.* AIAA Paper 94-1878, 1994; also AIAA Journal, 33 (1995), pp. 445-453.

[70] Braaten, M.E.; Connell, S.D.: *Three Dimensional Unstructured Adaptive Multigrid Scheme for the Navier-Stokes Equations.* AIAA Journal, 34 (1995), pp. 281-290.

[71] Luo, H.; Baum, J.D.; Löhner, R.; Cabello, J.: *Adaptive Edge-Based Finite Element Schemes for the Euler and Navier-Stokes Equations on Unstructured Grids.* AIAA Paper 93-0336, 1993.

[72] Crumpton, P.I.; Moiner, P.; Giles, M.B.: *An Unstructured Algorithm for High Reynolds Number Flows on Highly-Stretched Grids.* 10th Int. Conf. Num. Meth. for Laminar and Turbulent Flows, Swansea, England, July 21-25, 1997.

[73] Weiss, J.M.; Maruszewski, J.P.; Smith, W.A.: *Implicit Solution of Preconditioned Navier-Stokes Equations Using Algebraic Multigrid.* AIAA Journal, 37 (1999), pp. 29-36.

Chapter 6

Temporal Discretisation

The application of the *method of lines*, i.e., the separate spatial and temporal discretisation of the governing equations (2.19), leads, written down for each control volume, to a system of coupled ordinary differential equations in time

$$\frac{d\,(\Omega \bar{M} \vec{W})_I}{dt} = -\vec{R}_I\,. \tag{6.1}$$

In Eq. (6.1), Ω represents the volume, $\vec{R}$ the residual, $\bar{M}$ the mass matrix, and the index I denotes the particular control volume. The system (6.1) has to be integrated in time – either to obtain a steady-state solution ($\vec{R}_I = 0$), or to reproduce the time history of an unsteady flow.

We briefly discussed the aspects of the solution of the equation system (6.1) in Section 3.2. We saw that the various *explicit* and *implicit* methods can be derived from a basic non-linear scheme. It reads for a stationary grid

$$\frac{(\Omega \bar{M})_I}{\Delta t_I}\Delta \vec{W}_I^n = -\frac{\beta}{1+\omega}\,\vec{R}_I^{n+1} - \frac{1-\beta}{1+\omega}\,\vec{R}_I^n + \frac{\omega}{1+\omega}\frac{(\Omega \bar{M})_I}{\Delta t_I}\,\Delta \vec{W}_I^{n-1}\,, \tag{6.2}$$

where

$$\Delta \vec{W}_I^n = \vec{W}_I^{n+1} - \vec{W}_I^n \tag{6.3}$$

stands for the *update* (correction) of the solution. The superscripts n and $(n+1)$ denote the time levels. Hence, $\vec{W}^n$ means the flow solution at the present time t. Consequently, $\vec{W}^{n+1}$ represents the solution at the time $(t + \Delta t)$. The parameters β and ω determine the discretisation type (explicit or implicit) and also the temporal accuracy. For example, the condition expressed by Eq. (3.6) must be fulfilled to achieve second-order temporal accuracy.

In the following sections, we shall consider the most popular explicit and implicit time-stepping methods in some detail. We shall also present how the maximum allowable time step can be evaluated for a particular scheme. Furthermore, we shall discuss the issues of the appropriate implementations on structured as well as on unstructured grids. Finally, the last section will be devoted to time-accurate solutions of unsteady flow problems.

6.1 Explicit Time-Stepping Schemes

An explicit scheme starts from a known solution $\vec{W}^n$ and employs the corresponding residual $\vec{R}^n$ in order to obtain a new solution at time $(t + \Delta t)$. In other words, the new solution $\vec{W}^{n+1}$ depends solely on values already known. This fact makes the explicit schemes very simple and easy to implement.

As we discussed it in Subsection 3.2.1, a basic explicit scheme can be derived from Eq. (6.2) by setting $\beta = 0$ and $\omega = 0$. This results in

$$\bar{M}_I \, \Delta \vec{W}_I^n = -\frac{\Delta t_I}{\Omega_I} \, \vec{R}_I^n \, . \tag{6.4}$$

The mass matrix $\bar{M}$ can be lumped (i.e., substituted by the identity matrix) for steady problems or for the cell-centred discretisation.

The most popular and widespread explicit methods by far are the *multistage* (Runge-Kutta) time-stepping schemes and a variant, the *hybrid* multistage schemes. Therefore, we shall describe both methods in the following.

6.1.1 Multistage Schemes (Runge-Kutta)

The concept of explicit multistage schemes was first presented by Jameson et al. [1]. The multistage scheme advances the solution in a number of steps – so-called *stages* – which can be viewed as a sequence of updates according to Eq. (6.4). Applied to the discretised governing equations (6.1), where the mass matrix was lumped, an m-stage scheme reads

$$\begin{aligned}
\vec{W}_I^{(0)} &= \vec{W}_I^n \\
\vec{W}_I^{(1)} &= \vec{W}_I^{(0)} - \alpha_1 \frac{\Delta t_I}{\Omega_I} \vec{R}_I^{(0)} \\
\vec{W}_I^{(2)} &= \vec{W}_I^{(0)} - \alpha_2 \frac{\Delta t_I}{\Omega_I} \vec{R}_I^{(1)} \\
&\;\vdots \\
\vec{W}_I^{n+1} = \vec{W}_I^{(m)} &= \vec{W}_I^{(0)} - \alpha_m \frac{\Delta t_I}{\Omega_I} \vec{R}_I^{(m-1)} \, .
\end{aligned} \tag{6.5}$$

In the above expressions (6.5), α_k represents the stage coefficients. Furthermore, the denotation $\vec{R}_I^{(k)}$ means that the residual is evaluated with the solution $\vec{W}_I^{(k)}$ of the k-th stage.

Unlike in the classical Runge-Kutta schemes, only the zeroth solution and the last residual are stored here in order to reduce the memory requirements. The stage coefficients can be tuned to increase the maximum time step and to improve the stability for a particular spatial discretisation [2]-[4]. For consistency, it is only required that $\alpha_m = 1$. A consequence of the modification to the Runge-Kutta scheme is that second-order time accuracy can be realised only if $\alpha_{m-1} = 1/2$. Otherwise, the multistage scheme is first-order accurate in time.

	first-order scheme			second-order scheme		
stages	3	4	5	3	4	5
σ	1.5	2.0	2.5	0.69	0.92	1.15
α_1	0.1481	0.0833	0.0533	0.1918	0.1084	0.0695
α_2	0.4000	0.2069	0.1263	0.4929	0.2602	0.1602
α_3	1.0000	0.4265	0.2375	1.0000	0.5052	0.2898
α_4		1.0000	0.4414		1.0000	0.5060
α_5			1.0000			1.0000

Table 6.1: Multistage scheme: optimised stage coefficients (α) and CFL numbers (σ) for first- and second-order upwind spatial discretisations.

	central scheme $\sigma = 3.6$		upwind scheme $\sigma = 2.0$	
stage	α	β	α	β
1	0.2500	1.00	0.2742	1.00
2	0.1667	0.00	0.2067	0.00
3	0.3750	0.56	0.5020	0.56
4	0.5000	0.00	0.5142	0.00
5	1.0000	0.44	1.0000	0.44

Table 6.2: Hybrid multistage scheme: optimised stage (α) and blending (β) coefficients, as well as CFL numbers (σ) for central and upwind spatial discretisations.

The above multistage approach (6.5) is particularly suitable for upwind spatial discretisation on structured as well as unstructured grids. Central discretisation schemes perform more efficiently with the hybrid multistage methodology, which will be described next. Sets of optimised stage coefficients for first- and second-order upwind schemes are presented in Table 6.1 for three- to five-stage schemes [2]. Practical experience shows that the coefficients for the first-order scheme should be preferred in cases, where the flow field contains strong shocks, regardless of the order of the spatial discretisation. This can be explained by the fact that every higher-order scheme switches to first order at shocks to prevent oscillations of the solution. However, the residuals at strong shocks influence the convergence to steady state most significantly.

The main disadvantage of every explicit scheme is that the time step (Δt) is severely restricted by the characteristics of the governing equations as well as by the grid geometry. We shall discuss the computation of the maximum allowable time step in Subsection 6.1.4. Theoretical aspects of the determination of the time step and the so-called *CFL number* will be considered in Section 10.3 on stability analysis.

6.1.2 Hybrid Multistage Schemes

The computational work of an explicit multistage scheme (6.5), applied to the system (6.1), can be substantially reduced if the viscous fluxes and the dissipation are not re-evaluated at each stage. Additionally, the dissipation terms from different stages can be blended to increase stability of the scheme. Methods of this type were devised by Martinelli [5] and by Mavriplis et al. [6]. They are known as hybrid multistage schemes. Provided the stage coefficients are carefully optimised, the hybrid schemes are as robust as the basic multistage schemes.

For illustration, let us consider a popular 5-stage hybrid scheme, where the dissipative terms are evaluated at odd stages – generally denoted as the (5,3)-scheme. First, we split the spatial discretisation into two parts, i.e.,

$$\vec{R}_I = (\vec{R}_c)_I - (\vec{R}_d)_I \,. \tag{6.6}$$

The first part, $\vec{R}_c$, contains the central discretisation of the convective fluxes, which can be either the average of variables or the average of fluxes. It also includes the source term. The second part, $\vec{R}_d$, is composed of the viscous fluxes and the numerical dissipation. For example, in the case of the central scheme with artificial dissipation (Subsections 4.3.1 or 5.3.1) we would set

$$\begin{aligned}
(\vec{R}_c)_I &= \sum_{k=1}^{N_F} \left[\vec{F}_c(\vec{W}_{av}) \Delta S \right]_k - (\vec{Q}\Omega)_I \\
(\vec{R}_d)_I &= \sum_{k=1}^{N_F} \left[\vec{F}_v \, \Delta S + \vec{D} \right]_k \,,
\end{aligned}$$

where $\vec{W}_{av}$ represents the arithmetic average of flow variables from the left and the right side of face k.

With the residual split according to Eq. (6.6), the (5,3)-scheme can be formulated as

$$\begin{aligned}
\vec{W}_I^{(0)} &= \vec{W}_I^n \\
\vec{W}_I^{(1)} &= \vec{W}_I^{(0)} - \alpha_1 \frac{\Delta t_I}{\Omega_I} \left[\vec{R}_c^{(0)} - \vec{R}_d^{(0)} \right]_I \\
\vec{W}_I^{(2)} &= \vec{W}_I^{(0)} - \alpha_2 \frac{\Delta t_I}{\Omega_I} \left[\vec{R}_c^{(1)} - \vec{R}_d^{(0)} \right]_I \\
\vec{W}_I^{(3)} &= \vec{W}_I^{(0)} - \alpha_3 \frac{\Delta t_I}{\Omega_I} \left[\vec{R}_c^{(2)} - \vec{R}_d^{(2,0)} \right]_I \\
\vec{W}_I^{(4)} &= \vec{W}_I^{(0)} - \alpha_4 \frac{\Delta t_I}{\Omega_I} \left[\vec{R}_c^{(3)} - \vec{R}_d^{(2,0)} \right]_I \\
\vec{W}_I^{n+1} &= \vec{W}_I^{(0)} - \alpha_5 \frac{\Delta t_I}{\Omega_I} \left[\vec{R}_c^{(4)} - \vec{R}_d^{(4,2)} \right]_I \,,
\end{aligned} \tag{6.7}$$

where

$$\vec{R}_d^{(2,0)} = \beta_3 \vec{R}_d^{(2)} + (1-\beta_3)\vec{R}_d^{(0)}$$
$$\vec{R}_d^{(4,2)} = \beta_5 \vec{R}_d^{(4)} + (1-\beta_5)\vec{R}_d^{(2,0)}\,. \tag{6.8}$$

The stage coefficients α_m and the blending coefficients β_m in the above relations (6.7), (6.8) are given in Table 6.2 for central and upwind schemes. Both sets of coefficients are particularly optimised for the multigrid method (Section 9.4). We shall discuss the properties of the above hybrid multistage scheme later in Section 10.3.

It should be mentioned that it is also popular to evaluate the dissipation term $\vec{R}_d$ in the first two stages only, without any blending. A well-known (5,2)-scheme, which is often employed with the central spatial discretisation, uses the stage coefficients of Table 6.2. However, the (5,2)-scheme is less suitable for multigrid than the above (5,3)-scheme.

6.1.3 Treatment of the Source Term

There are certain cases in which the source term $\vec{Q}$ in Eq. (4.2) or (5.2) becomes dominant. Such situation is often encountered when chemistry or turbulence models are employed. The problem is that a large source term changes the flow variables rapidly in space and in time. The changes due to a strong source term happen at much smaller time scales than those of the flow equations. This increases the *stiffness* of the governing equations significantly. The stiffness is defined as the ratio of the largest to the smallest eigenvalue of the Jacobian matrix $\partial\vec{R}/\partial\vec{W}$. The stiffness can also be viewed as the ratio of the largest to the smallest time scale.

When we apply one of the above explicit multistage schemes (or any other purely explicit scheme) to a stiff system of equations, we will have to reduce the time step considerably in order to stabilise the time integration. Hence, the convergence to the steady state will become very slow. More seriously, an explicit scheme can fail to find the correct solution [7]. A remedy suggested by Curtiss et al. [8] is to treat the source term in an implicit way. In order to demonstrate the approach, we rewrite the basic explicit scheme in Eq. (6.4) as follows (cf. Eq. (4.2) or (5.2))

$$\frac{\Omega_I}{\Delta t_I}\Delta\vec{W}_I^n = -\left[\sum_{k=1}^{N_F}(\vec{F}_c^n - \vec{F}_v^n)_k\,\Delta S_k \;-\; \Omega_I\vec{Q}_I^{n+1}\right], \tag{6.9}$$

where the source term is now evaluated at the new time level $(n{+}1)$. For simplicity, the mass matrix was omitted from Eq. (6.9). Since the value of the source term at the time $(n{+}1)$ is unknown, we have to approximate it. For this purpose, we linearise the source term about the current time level n, resulting in

$$\vec{Q}^{n+1} \approx \vec{Q}^n + \frac{\partial\vec{Q}}{\partial\vec{W}}\Delta\vec{W}^n\,. \tag{6.10}$$

If we insert Eq. (6.10) into Eq. (6.9) and rearrange the terms, we obtain the following relation [9], [10]

$$\left[\frac{\bar{I}}{\Delta t_I} - \left(\frac{\partial \vec{Q}}{\partial \vec{W}}\right)_I\right] \Delta \vec{W}_I^n = -\frac{1}{\Omega_I}\,\vec{R}_I^n\,, \tag{6.11}$$

where $\bar{I}$ represents the identity matrix. The formulation (6.11) is called *point implicit* because the term in square brackets on the left-hand side – the implicit operator – depends only on values in the control volume Ω_I itself. A comparison with Eq. (6.9) reveals that the scalar time step Δt changed now to a matrix. Thus, each flow equation becomes scaled by an individual parameter, corresponding to the associated eigenvalue. In this way, the disparity between the time scales is offset and the time step restriction due to the source term is alleviated.

When we apply the above point-implicit approach to the multistage scheme in Eq. (6.5), we obtain for the k-th stage

$$\vec{W}_I^{(k)} = \vec{W}_I^{(0)} - \frac{\alpha_k}{\Omega_I}\left[\frac{1}{\Delta t_I} - \left(\frac{\partial \vec{Q}}{\partial \vec{W}}\right)_I^{(k)}\right]^{-1} \vec{R}_I^{(k-1)}\,. \tag{6.12}$$

A similar expression holds also for the hybrid multistage scheme in Eq. (6.7). The interested reader may find a detailed investigation of the influence of the source term on stability in Refs. [11]-[13].

6.1.4 Determination of the Maximum Time Step

Every explicit time-stepping scheme remains stable only up to a certain value of the time step Δt. To be stable, a time-stepping scheme has to fulfil the so-called *Courant-Friedrichs-Lewy (CFL)* condition [14]. It states that the domain of dependence of the numerical method has to include the domain of dependence of the partial differential equation. The CFL condition means for the basic explicit scheme (6.4) that the time step should be equal to or smaller than the time required to transport information across the stencil of the spatial discretisation scheme. Hence, in 1D the condition for the time step would read for the linear convection equation

$$\Delta t = \sigma\,\frac{\Delta x}{|\Lambda_c|}\,, \tag{6.13}$$

where $\Delta x/|\Lambda_c|$ represents the time necessary to propagate information over the cell size Δx with the velocity Λ_c. The velocity Λ_c corresponds to the maximum eigenvalue of the convective flux Jacobian. The positive coefficient σ denotes the *CFL number*. The magnitude of the CFL number depends on the type and the parameters of the time-stepping scheme, as well as on the form of the spatial discretisation scheme. We shall investigate the dependency of σ in Section 10.3 for two model problems. Tables 6.1 and 6.2 list the CFL numbers for various multistage schemes and discretisations.

The maximum time step can be determined for linear model equations with the aid of Von Neumann stability analysis (Section 10.3). However, the maximum time step can be calculated only approximately in multiple dimensions and for non-linear governing equations. In the following, we will present relations for the estimation of the time step on structured and unstructured grids for inviscid as well as for viscous flows.

Time Step on Structured Grids

<u>Euler Equations</u>

On a structured grid, the time step Δt can be determined for a control volume Ω_I from the approximate relation [15]-[17]

$$\Delta t_I = \sigma \frac{\Omega_I}{(\hat{\Lambda}_c^I + \hat{\Lambda}_c^J + \hat{\Lambda}_c^K)_I} . \tag{6.14}$$

The CFL number σ can be obtained for multistage schemes from Table 6.1 and for hybrid schemes from Table 6.2. The spectral radii of the convective flux Jacobians (A.7) read for the three grid directions

$$\begin{aligned} \hat{\Lambda}_c^I &= \left(|\vec{v}\cdot\vec{n}^I| + c\right)\Delta S^I \\ \hat{\Lambda}_c^J &= \left(|\vec{v}\cdot\vec{n}^J| + c\right)\Delta S^J \\ \hat{\Lambda}_c^K &= \left(|\vec{v}\cdot\vec{n}^K| + c\right)\Delta S^K . \end{aligned} \tag{6.15}$$

The normal vectors and face areas in Eq. (6.15) are obtained by averaging the corresponding values from the two opposite sides of the control volume in the respective direction. For example, if a dual control volume (Subsection 4.2.3) would be oriented as sketched in Fig. 4.1b, we would use in the I-direction

$$\vec{n}^I_{i,j,k} = \frac{1}{2}(\vec{n}_1 - \vec{n}_2) , \quad \Delta S^I_{i,j,k} = \frac{1}{2}(\Delta S_1 + \Delta S_2) . \tag{6.16}$$

Similar expressions hold for the J- and K-direction.

With Eq. (6.14), we obtain a *local* time step, which is valid for one control volume only. If we are interested in a steady state solution, we may use the local time step to accelerate the convergence (cf. Section 9.1). However, if time accuracy is important, we have to employ one *global* time step for all volumes, i.e.,

$$\Delta t = \min_I (\Delta t_I) , \tag{6.17}$$

where a minimum over all control volumes is taken.

Navier-Stokes Equations

For viscous flows, the spectral radii of the viscous flux Jacobians (A.8) have to be included in the computation of Δt. They can severely limit the maximum time step in boundary layers. The time step can be evaluated from [5], [16], [17]

$$\Delta t_I = \sigma \frac{\Omega_I}{(\hat{\Lambda}_c^I + \hat{\Lambda}_c^J + \hat{\Lambda}_c^K)_I + C(\hat{\Lambda}_v^I + \hat{\Lambda}_v^J + \hat{\Lambda}_v^K)_I} . \tag{6.18}$$

The constant which multiplies the viscous spectral radii is usually set as $C = 4$. If we assume that an eddy-viscosity turbulence model is employed, the viscous spectral radii are given by [16], [17]

$$\Lambda_v^I = \max\left(\frac{4}{3\rho}, \frac{\gamma}{\rho}\right)\left(\frac{\mu_L}{Pr_L} + \frac{\mu_T}{Pr_T}\right)\frac{(\Delta S^I)^2}{\Omega} \tag{6.19}$$

and similarly for the other directions. In Equation (6.19), μ_L denotes the laminar and μ_T the turbulent dynamic viscosity coefficient, respectively. Furthermore, Pr_L and Pr_T are the laminar and the turbulent Prandtl numbers. The CFL numbers in Tables 6.1 and 6.2 apply also for viscous flows. Particularly efficient for viscous flows is the (5,3) hybrid scheme from Eq. (6.7).

Time Step on Unstructured Grids

Several approaches were suggested for the estimation of the maximum time step on unstructured grids. We shall present two different approaches below.

Method 1

One proven method, which closely follows the implementation on structured grids, reads [6]

$$\Delta t_I = \sigma \frac{\Omega_I}{(\hat{\Lambda}_c + C\hat{\Lambda}_v)_I} , \tag{6.20}$$

where $\hat{\Lambda}_c$ and $\hat{\Lambda}_v$ represent a sum of the convective and viscous spectral radii over all faces of the control volume. As on structured grids, $C = 4$ is usually used. In the case of the cell-centred scheme, the spectral radii are defined as [6]

$$\begin{aligned}(\hat{\Lambda}_c)_I &= \sum_{J=1}^{N_F}\left(|\vec{v}_{IJ} \cdot \vec{n}_{IJ}| + c_{IJ}\right)\Delta S_{IJ} \\ (\hat{\Lambda}_v)_I &= \frac{1}{\Omega_I}\sum_{J=1}^{N_F}\left[\max\left(\frac{4}{3\,\rho_{IJ}}, \frac{\gamma_{IJ}}{\rho_{IJ}}\right)\left(\frac{\mu_L}{Pr_L} + \frac{\mu_T}{Pr_T}\right)_{IJ}(\Delta S_{IJ})^2\right] .\end{aligned} \tag{6.21}$$

The values of the flow variables at the faces of the control volume are obtained by arithmetic averaging.

Method 2

The spectral radii predicted by Eq. (6.21) are too large, particularly on mixed element grids. This leads to smaller time step than necessary. The implementation in Ref. [18] offers a more accurate estimation of the time step, namely

$$\Delta t_I = \sigma \frac{\Omega_I}{(\hat{\Lambda}_c^x + \hat{\Lambda}_c^y + \hat{\Lambda}_c^z)_I + C(\hat{\Lambda}_v^x + \hat{\Lambda}_v^y + \hat{\Lambda}_v^z)_I} \tag{6.22}$$

with the convective spectral radii

$$\begin{aligned} \hat{\Lambda}_c^x &= (|u| + c)\, \Delta\hat{S}^x \\ \hat{\Lambda}_c^y &= (|v| + c)\, \Delta\hat{S}^y \\ \hat{\Lambda}_c^z &= (|w| + c)\, \Delta\hat{S}^z \end{aligned} \tag{6.23}$$

and with the viscous spectral radii (eddy-viscosity turbulence model assumed)

$$\Lambda_v^x = \max\left(\frac{4}{3\rho}, \frac{\gamma}{\rho}\right)\left(\frac{\mu_L}{Pr_L} + \frac{\mu_T}{Pr_T}\right)\frac{(\Delta S^x)^2}{\Omega}\,, \text{ etc.} \tag{6.24}$$

The variables $\Delta\hat{S}^x$, $\Delta\hat{S}^y$ and $\Delta\hat{S}^z$, respectively, represent projections of the control volume on the y-z-, x-z- and the x-y-plane. They are given by the formulae

$$\begin{aligned} \Delta\hat{S}^x &= \frac{1}{2}\sum_{J=1}^{N_F} |S_x|_J \\ \Delta\hat{S}^y &= \frac{1}{2}\sum_{J=1}^{N_F} |S_y|_J \\ \Delta\hat{S}^z &= \frac{1}{2}\sum_{J=1}^{N_F} |S_z|_J\,, \end{aligned} \tag{6.25}$$

where S_x, S_y and S_z denote the x-, y- and the z-component of the face vector $\vec{S} = \vec{n} \cdot \Delta S$.

The CFL numbers stay in general the same as on structured grids. Thus, the values collected in Tables 6.1 and 6.2 still apply. Furthermore, the convergence to steady state can also be accelerated by local time stepping, in the same way as on the structured grids. A global time step, necessary for simulating unsteady flows, can be obtained from Eq. (6.17) as before.

6.2 Implicit Time-Stepping Schemes

Various implicit time integration schemes can be obtained by setting $\beta \neq 0$ in Eq. (6.2). An implicit scheme with $\omega = 0$ was found to be best suited for the solution of stationary flow problems (for unsteady flows see Section 6.3). Herewith, Eq. (6.2) simplifies to

$$\frac{(\Omega \bar{M})_I}{\Delta t_I} \Delta \vec{W}_I^n = -\beta \, \vec{R}_I^{n+1} - (1-\beta) \, \vec{R}_I^n \, . \tag{6.26}$$

As we can see, the implicit formulation leads to a set of non-linear equations for the unknown flow variables at the time $(t+\Delta t)$. The solution of Eq. (6.26) requires the evaluation of the residual at the new time level, i.e., $\vec{R}^{n+1}$. Since we do not know $\vec{W}^{n+1}$, this cannot be done directly. However, we can linearise the residual $\vec{R}^{n+1}$ in Eq. (6.26) about the current time level, i.e.,

$$\vec{R}_I^{n+1} \approx \vec{R}_I^n + \left(\frac{\partial \vec{R}}{\partial \vec{W}} \right)_I \Delta \vec{W}^n \, , \tag{6.27}$$

where the term $\partial \vec{R} / \partial \vec{W}$ is referred to as the *flux Jacobian*. We should mention that the flux Jacobian is often derived from a rather crude approximation to the spatial discretisation represented by $\vec{R}^n$. For example, in the case of higher-order upwind discretisations, it is quite common to base the flux Jacobian solely on a first-order upwind scheme. However, for best efficiency and robustness, the flux Jacobian should still reflect the most important features of the spatial discretisation.

If we substitute now the linearisation in Eq. (6.27) for $\vec{R}^{n+1}$ in Eq. (6.26), we obtain the following implicit scheme

$$\left[\frac{(\Omega \bar{M})_I}{\Delta t_I} + \beta \left(\frac{\partial \vec{R}}{\partial \vec{W}} \right)_I \right] \Delta \vec{W}^n = -\vec{R}_I^n \, , \tag{6.28}$$

The term in square brackets on the left-hand side of Eq. (6.28) is referred to as the *implicit operator* or the *system matrix*. Consequently, the right-hand side of Eq. (6.28) is called the *explicit operator*. It is only the explicit operator that determines the spatial accuracy of the solution.

The implicit operator constitutes a large, sparse, and non-symmetric matrix with dimensions equal to the total number of cells (cell-centred scheme) or grid points (cell-vertex scheme). Below we will discuss further the form of the implicit operator for structured as well as for unstructured grids. As we already saw in Section 3.2, the mass matrix $\bar{M}$ can be replaced by the identity matrix, without influencing the steady state solution. The parameter β in Eq. (6.28) is generally set to 1, which results in a 1st-order accurate temporal discretisation. A 2nd-order time accurate scheme is obtained for $\beta = 1/2$. However, this is not recommended since the scheme with $\beta = 1$ is much more robust, and the time accuracy is of no importance for steady problems.

In the case of stiff governing equations, the source term has to be included in the implicit operator. This happens quite naturally with the linearisation of the residual in Eq. (6.27), which leads to a formulation identical to Eq. (6.10). As demonstrated in Ref. [12], the above implicit scheme (6.28) remains stable for any time step if the eigenvalues of $\partial\vec{Q}/\partial\vec{W}$ are all negative or zero.

The solution of the linear equation system (6.28) requires the inversion of the implicit operator, i.e., the inversion of a very large matrix. In principle, this can be done in two ways. The first one consists of a direct matrix inversion, using either the Gaussian elimination or some direct sparse matrix method [19], [20]. However, because of the excessive amount of memory and a very high computational effort, this approach is not suited for practical problems [21].

The second possibility of inverting the implicit operator represent iterative methods. We mentioned the most widely used ones in Subsection 3.2.2. Iterative methods can be divided roughly into two groups. The first one consists of approaches which decompose the implicit operator into several parts – a process called *factorisation*. The factors are constructed such that they can be more easily inverted than the original implicit operator. To the second group belong schemes, which employ a Krylov-subspace method for the inversion of the implicit operator. In this case, the implicit time-stepping scheme (6.28) is usually turned into Newton's method by setting $\Delta t \to \infty$. The scheme is then named *Newton-Krylov* method.

In the following, we shall discuss first the matrix structure of the implicit operator. Then we shall investigate the possibilities of computing the flux Jacobian $\partial\vec{R}/\partial\vec{W}$ in Eq. (6.28). Finally, we shall present the three most popular iterative methods in detail.

6.2.1 Matrix Form of Implicit Operator

Referring to Eq. (6.27), we can write the linearisation of the residual $\vec{R}^{n+1}$ in the form

$$\begin{aligned} \vec{R}^{n+1} \approx \vec{R}^{n} + \sum_{m=1}^{N_F} \left\{ \frac{\partial}{\partial\vec{W}} \left[(\vec{F}_c - \vec{F}_v)_m \, \Delta S_m \right] \Delta\vec{W}^{n} \right\} \\ - \frac{\partial(\Omega\vec{Q})}{\partial\vec{W}} \Delta\vec{W}^{n} \end{aligned} \tag{6.29}$$

with N_F being the number of faces of the control volume Ω (cf. Eq. (4.2) or (5.2)). Thus, the flux Jacobian reads

$$\frac{\partial\vec{R}}{\partial\vec{W}} = \sum_{m=1}^{N_F} \frac{\partial(\vec{F}_c)_m}{\partial\vec{W}} \Delta S_m - \sum_{m=1}^{N_F} \frac{\partial(\vec{F}_v)_m}{\partial\vec{W}} \Delta S_m - \frac{\partial(\Omega\vec{Q})}{\partial\vec{W}} . \tag{6.30}$$

It should be stressed that the flux Jacobian has to be conceived as an operator which acts on the update $\Delta\vec{W}$. As stated above, the convective and viscous fluxes in Eq. (6.30) do not necessarily need to be identical to the fluxes in the explicit operator.

Because of significant differences between the implicit operator on structured and unstructured grids, we shall treat each case separately.

Implicit Operator on Structured Grids

In order to derive the form of the system matrix in Eq. (6.28), let us consider the 1-D grid in Fig. 6.1. Let us further assume that the cell-vertex scheme with dual control volumes (Subsection 4.2.3) and a simple average of fluxes are used for the spatial discretisation (cf. Fig. 4.8). In absence of viscous fluxes, the residual is given by

$$\vec{R}_i^n = (\vec{F}_c)_{i+1/2}\,\Delta S_{i+1/2} + (\vec{F}_c)_{i-1/2}\,\Delta S_{i-1/2} - \Omega_i \vec{Q}_i\,. \tag{6.31}$$

Furthermore, the derivative of the convective fluxes at the face $m = i + 1/2$ in Eq. (6.30) can be expressed as follows

$$\begin{aligned} \frac{\partial (\vec{F}_c)_{i+1/2}}{\partial \vec{W}} &= \frac{\partial}{\partial \vec{W}} \left\{ \frac{1}{2} \left[\vec{F}_c(\vec{W}_{i+1}^n) + \vec{F}_c(\vec{W}_i^n) \right] \right\} \\ &= \frac{1}{2} \left[(\bar{A}_c)_{i+1} + (\bar{A}_c)_i \right], \end{aligned} \tag{6.32}$$

where $\bar{A}_c$ denotes the convective flux Jacobian (Section A.7). Hence, according to Eqs. (6.29), (6.31), and (6.32), the residual at $(t+\Delta t)$ is approximated as

$$\begin{aligned} \vec{R}_i^{n+1} \approx \vec{R}_i^n &+ \frac{1}{2} \left[(\bar{A}_c)_{i+1}\, \Delta \vec{W}_{i+1}^n + (\bar{A}_c)_i\, \Delta \vec{W}_i^n \right] \Delta S_{i+1/2} \\ &+ \frac{1}{2} \left[(\bar{A}_c)_{i-1}\, \Delta \vec{W}_{i-1}^n + (\bar{A}_c)_i\, \Delta \vec{W}_i^n \right] \Delta S_{i-1/2} \\ &- \frac{\partial (\Omega_i \vec{Q}_i)}{\partial \vec{W}}\,. \end{aligned} \tag{6.33}$$

Finally, for $\beta = 1$ and a lumped mass matrix we can derive from Eq. (6.26) the implicit scheme

$$\begin{aligned} \Bigg\{ \frac{\Omega_i}{\Delta t_i} \bar{I} &+ \frac{1}{2} \left[(\bar{A}_c)_i\, \Delta S_{i+1/2} + (\bar{A}_c)_i\, \Delta S_{i-1/2} \right] \\ &- \frac{\partial (\Omega_i \vec{Q}_i)}{\partial \vec{W}} + \frac{1}{2} \left[(\bar{A}_c)_{i+1}\, \Delta S_{i+1/2} \right] \\ &+ \frac{1}{2} \left[(\bar{A}_c)_{i-1}\, \Delta S_{i-1/2} \right] \Bigg\} \Delta \vec{W}^n = -\vec{R}_i^n\,, \end{aligned} \tag{6.34}$$

where $\bar{I}$ stands for the identity matrix. It should be noted that the unit normal vector in each matrix $\bar{A}_c$ is evaluated at the same side of the control volume as the associated area ΔS. As we can see, the implicit operator involves the same 3-point stencil (nodes $i-1$, i, and $i+1$) as the spatial discretisation. In

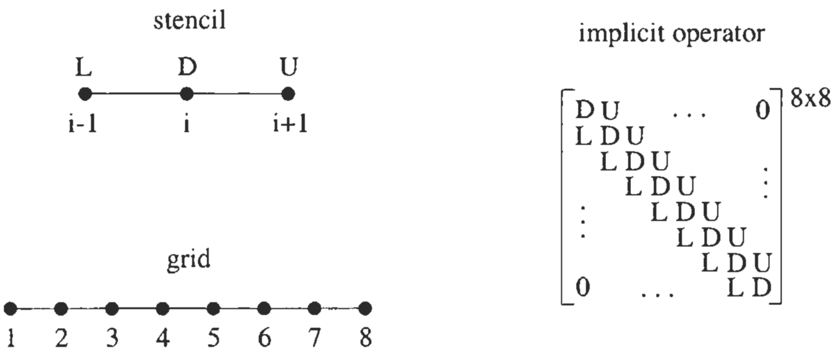


Figure 6.1: 1-D structured grid and the associated implicit operator matrix for a 3-point stencil.

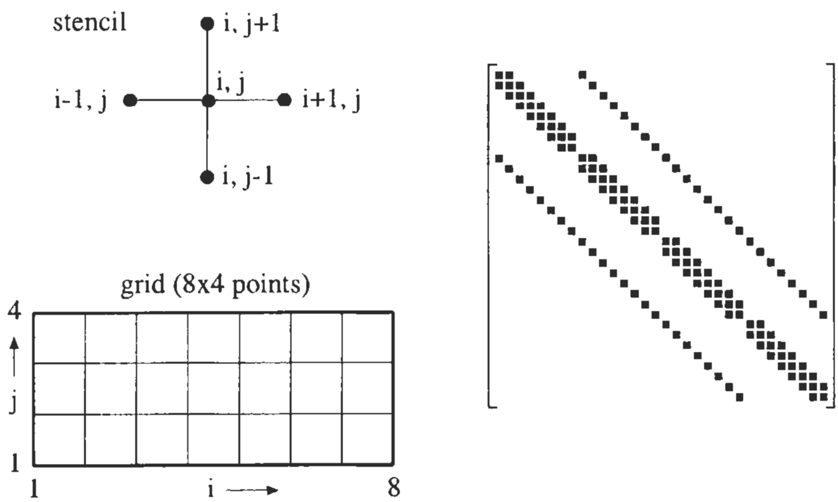


Figure 6.2: 2-D structured grid (left) and the associated implicit operator matrix for a 5-point stencil (right). Nonzero block matrices displayed as filled rectangles.

order to visualise the system matrix, we denote all terms in the implicit operator associated with the central node i as $\mathbf{D}$, i.e.,

$$\mathbf{D} \equiv \frac{\Omega_i}{\Delta t_i}\bar{I} + \frac{1}{2}\left[(\bar{A}_c)_i\,\Delta S_{i+1/2} + (\bar{A}_c)_i\,\Delta S_{i-1/2}\right] - \frac{\partial(\Omega_i\vec{Q}_i)}{\partial\vec{W}}\,,$$

with the downwind node $(i+1)$ as $\mathbf{U}$, i.e.,

$$\mathbf{U} \equiv \frac{1}{2}\,(\bar{A}_c)_{i+1}\,\Delta S_{i+1/2}\,,$$

and with the upwind node $(i-1)$ as $\mathbf{L}$, i.e.,

$$\mathbf{L} \equiv \frac{1}{2}\,(\bar{A}_c)_{i-1}\,\Delta S_{i-1/2}\,,$$

respectively. Writing down Eq. (6.34) for all eight nodes of the grid in Fig. 6.1, we obtain the 8×8 block-tridiagonal matrix displayed on the right side of Fig. 6.1. Each of the blocks $\mathbf{L}$, $\mathbf{D}$, and $\mathbf{U}$ represents a 3×3 matrix in 1D (because of the three conservation equations).

The same ideas carry over to multiple dimensions. For example, if the spatial discretisation in 2D would involve the 5-point stencil sketched in Fig. 6.2, we would obtain a block-pentadiagonal matrix. This is shown on the right side of Fig. 6.2. The nodes were ordered such that the i-index runs faster than the j-index (corresponds to $mat(i,j)$ in FORTRAN). It should be noted that the second off-diagonal is at the distance of eight (total number of nodes in i-direction) elements from the main diagonal. Finally, the system matrix would become block-septadiagonal in 3D, if we would employ the 7-point stencil of Fig. 4.9b for the spatial discretisation.

In summary, we can state that the system matrix always possesses a regular, sparse and banded form for structured grids. It should be further mentioned that the term

$$(\Omega\bar{M})_I/\Delta t_I \tag{6.35}$$

in Eq. (6.28), which is always located on the main diagonal, can cause difficulties. Namely, if the time step becomes large, some iterative matrix inversion schemes (e.g., Gauss-Seidel) may fail due to reduced diagonal dominance of the implicit operator.

Implicit Operator on Unstructured Grids

The appearance of the system matrix changes completely when we proceed from structured to unstructured grids. This can be demonstrated with the aid of a small unstructured grid sketched in Fig. 6.3. We want assume that the spatial discretisation is given by the cell-centred scheme presented in Subsection 5.2.1, which uses flow values only from the nearest neighbours (e.g., like a first-order upwind scheme – see Subsections 5.3.2 and 5.3.3). Thus, for example, the stencil for cell 2 includes the cells 18, 10, and 13. The resulting system matrix is displayed on the right side of Fig. 6.3. It is obvious that since the grid cells

(nodes in the case of a median-dual scheme) are in general numbered in an arbitrary order, no regular pattern can be expected for the matrix. Only the main diagonal, which contains at least the expression (6.35) and possibly the derivative of the source term, is always present.

The quasi-random distribution of nonzero elements in the system matrix is undesirable. It slows down the convergence of iterative inversion methods like Gauss-Seidel. Furthermore, preconditioning techniques for Krylov-subspace methods like ILU (Incomplete Lower-Upper) factorisation scheme cannot be used efficiently. Therefore, strategies were developed where the cells (nodes) are renumbered such that the bandwidth of the system matrix is considerably reduced, i.e., the nonzero elements are clustered close to the main diagonal. The best-known renumbering strategy is the *Reverse-Cuthill-McKee* (RCM) algorithm [22], [23]. Figure 6.4 shows the resulting cell numbering and the system matrix when the RCM algorithm is applied to the example grid of Fig. 6.3. As we can see, the bandwidth of the matrix is significantly reduced and the matrix obtains a more regular structure.

Several other renumbering strategies were developed in order to minimise cache misses or to allow for vectorisation of the numerical scheme. An overview can be found, e.g., in Ref. [24].

6.2.2 Evaluation of the Flux Jacobian

Depending on the type of the underlying spatial discretisation scheme, an analytical evaluation of the flux Jacobian $\partial\vec{R}/\partial\vec{W}$ in Eq. (6.28) may become very complex if not impossible. In order to make the concepts more clear, we shall derive the flux Jacobian for inviscid flows and then discuss the extension to the Navier-Stokes equations.

Central Scheme

The flux Jacobian is most easily formulated in the case of the central spatial discretisation. As we already saw for the example in Fig. 6.1, $\partial\vec{R}/\partial\vec{W}$ consists of convective flux Jacobians (cf. Eq. (6.34)), which can be derived analytically (see Section A.7). Artificial viscosity is usually included in a simplified form, without the non-linear pressure sensor (Eq. (4.55)). We shall return to this below in Subsection 6.2.3.

Flux-Vector Splitting Scheme

The evaluation of the flux Jacobian becomes more involved when one of the flux-vector splitting schemes (Subsection 4.3.2) is used as the basis for its derivation. Let us, for illustration, consider the scheme due to Steger and Warming [25]. Previous investigations [26] revealed that the Steger-Warming splitting is preferable over, e.g., the Van Leer's flux-vector splitting scheme (Eq. (4.60)) for various upwind discretisations of the explicit operator.

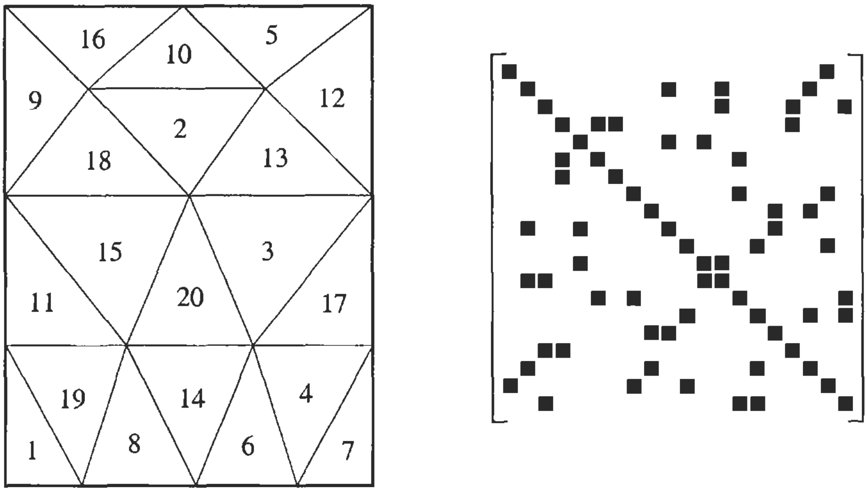


Figure 6.3: 2-D unstructured grid (left) and the associated implicit operator matrix for a nearest neighbour stencil (right). Nonzero block matrices displayed as filled rectangles.

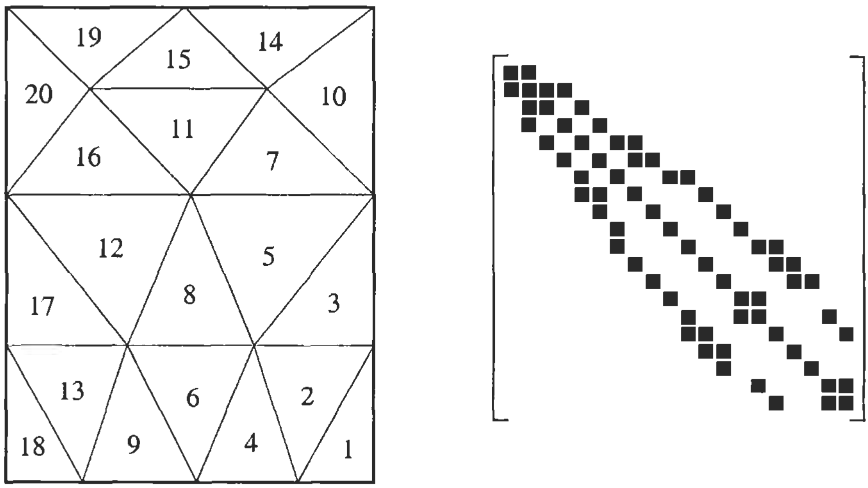


Figure 6.4: Reduced bandwidth (from 18 to 5) of the implicit operator from Fig. 6.3 with reverse-Cuthill-McKee ordering. Nonzero block matrices displayed as filled rectangles.

The basic idea of the Steger-Warming flux-vector splitting scheme is to split the convective fluxes into a positive and negative part, i.e.,

$$\vec{F}_c = \vec{F}_c^+ + \vec{F}_c^- \tag{6.36}$$

with the fluxes defined as

$$\vec{F}_c^\pm = \bar{A}_{SW}^\pm \vec{W} = \left(\bar{M}\bar{\Lambda}^\pm \bar{M}^{-1}\right)\vec{W} \,. \tag{6.37}$$

In Eq. (6.37), $\bar{A}_{SW}^\pm$ denotes the positive/negative Steger-Warming flux-splitting Jacobian. Furthermore, $\bar{M}$ represents the matrix of right eigenvectors, $\bar{M}^{-1}$ the matrix of left eigenvectors, and $\bar{\Lambda}^\pm$ stands for the diagonal matrix of positive/negative eigenvalues, respectively (cf. Section A.9). The eigenvalue matrices are defined as [25]

$$\bar{\Lambda}^\pm = \frac{1}{2}(\bar{\Lambda}_c \pm |\bar{\Lambda}_c|) \,, \tag{6.38}$$

where $\bar{\Lambda}_c$ is given by Eq. (A.63).

Using the splitting defined in Eq. (6.36), we obtain for the product of the flux Jacobian with the update $\Delta\vec{W}^n$ in Eq. (6.29)

$$\frac{\partial \vec{R}_I}{\partial \vec{W}}\Delta\vec{W}^n = \sum_{m=1}^{N_F}\left[\frac{\partial(\vec{F}_c^+\Delta S)_m}{\partial \vec{W}_{L,m}}\Delta\vec{W}_{L,m}^n + \frac{\partial(\vec{F}_c^-\Delta S)_m}{\partial \vec{W}_{R,m}}\Delta\vec{W}_{R,m}^n\right] \,. \tag{6.39}$$

In the above Eq. (6.39), $\Delta\vec{W}_{L,m}^n$ and $\Delta\vec{W}_{R,m}^n$ denote the updates of the left and right state at the face m, respectively. On structured grids, the left and right state can be evaluated by the MUSCL approach (Eq. (4.46)). On unstructured grids, the reconstruction methods discussed in Subsection 5.3.3 can be applied. However, the stencil becomes wider with increasing accuracy, which leads to larger bandwidth of the system matrix. Therefore, and in order to reduce the numerical complexity, only first-order accurate approximation is usually employed in Eq. (6.39). As a compromise, we could reconstruct the left and right state with higher accuracy but retain the stencil of the first-order scheme for the evaluation of the derivatives [27].

To proceed with the discussion on the evaluation of the derivatives $\partial\vec{F}_c^\pm/\partial\vec{W}$ in Eq. (6.39), let us consider, e.g., the positive flux at face m

$$\frac{\partial(\vec{F}_c^+\Delta S)_m}{\partial \vec{W}_{L,m}} = \frac{\partial}{\partial \vec{W}_{L,m}}\left[(\bar{A}_{SW}^+\vec{W})_{L,m}\,\Delta S_m\right]$$

which becomes with Eqs. (6.37), (6.38)

$$\begin{aligned}\frac{\partial(\vec{F}_c^+\Delta S)_m}{\partial \vec{W}_{L,m}} &= \frac{\Delta S_m}{2}\left[(\bar{A}_c)_{L,m} + |(\bar{A}_c)_{L,m}|\right] \\ &+ \frac{\Delta S_m}{2}\left[\frac{\partial(\bar{A}_c)_{L,m}}{\partial \vec{W}_{L,m}} + \frac{\partial|(\bar{A}_c)_{L,m}|}{\partial \vec{W}_{L,m}}\right]\vec{W}_{L,m}^n \,.\end{aligned} \tag{6.40}$$

An expression similar to Eq. (6.40) can also be found for the negative flux. As we can see, the first term in Eq. (6.40) consists of convective flux Jacobians (see Section A.7) and thus presents no difficulty. However, the second term involves derivatives of matrix elements. Although it is possible to obtain the derivatives analytically either by hand calculation or by using a symbolic algebra package, this will produce a large, computationally inefficient code [28]. Alternatively, it is possible to assume the matrix $\bar{A}_c$ is locally constant so that the second term in Eq. (6.40) can be neglected. However, depending on the type of the implicit scheme, this may severely restrict the CFL number [29].

Other approaches that we could use to compute the derivatives $\partial \vec{F}_c^{\pm}/\partial \vec{W}$ in Eq. (6.39) would be the automatic differentiation of the source code (e.g., using ADIFOR [30]) or the finite-difference method (see, e.g., [21], [28]). Herewith, the derivative of the i-th component of a vector $\vec{F}$ with respect to the j-th component of a dependent variable $\vec{X}$ can be approximated as

$$\frac{\partial f_i}{\partial x_j} \approx \frac{f_i(\vec{X} + h_j \vec{e}^{\,j}) - f_i(\vec{X})}{h_j}\,, \tag{6.41}$$

where $\vec{e}^{\,j}$ denotes the j-th standard basis vector. Dennis and Schnabel [31] suggested a stepsize h_j of the form

$$h_j = \sqrt{\epsilon}\, \max\{|x_j|, \operatorname{typ} x_j\}\, \operatorname{sign}(x_j) \tag{6.42}$$

with ϵ being the machine accuracy and $\operatorname{typ} x_j$ a typical size of x_j.

The reader is also referred to [32] and [33] for hints on efficient numerical evaluation of Jacobian matrices.

Flux-Difference Splitting Scheme

In the case of the flux-difference splitting scheme due to Roe (Subsection 4.3.3, Eq. (4.87)), we can write the product of the flux Jacobian with the update in Eq. (6.29) as

$$\begin{aligned}
\frac{\partial \vec{R}_I}{\partial \vec{W}} \Delta \vec{W}^n = \sum_{m=1}^{N_F} \frac{\Delta S_m}{2} \Big\{ & (\bar{A}_c)_{L,m} \Delta \vec{W}^n_{L,m} + (\bar{A}_c)_{R,m} \Delta \vec{W}^n_{R,m} \\
& - \frac{\partial}{\partial \vec{W}_{L,m}} \left[|\bar{A}_{Roe}|_m (\vec{W}^n_{R,m} - \vec{W}^n_{L,m}) \right] \Delta \vec{W}^n_{L,m} \\
& - \frac{\partial}{\partial \vec{W}_{R,m}} \left[|\bar{A}_{Roe}|_m (\vec{W}^n_{R,m} - \vec{W}^n_{L,m}) \right] \Delta \vec{W}^n_{R,m} \Big\}\,.
\end{aligned} \tag{6.43}$$

Similar to flux-vector splitting, the expression (6.43) contains convective flux Jacobians as well as derivatives of the Roe matrix $\bar{A}_{Roe}$. The derivatives were presented in [29]. Since the corresponding formulae are very complex (see also Ref. [28]), it is a better idea to evaluate the term $\partial \vec{F}_c/\partial \vec{W}$ numerically, as

discussed above. However, we can also simplify Eq. (6.43) by assuming locally constant Roe matrices [34]

$$\frac{\partial \vec{R}_I}{\partial \vec{W}} \Delta \vec{W}^n \approx \sum_{m=1}^{N_F} \frac{\Delta S_m}{2} \Big\{ (\bar{A}_c)_{L,m} \Delta \vec{W}^n_{L,m} + (\bar{A}_c)_{R,m} \Delta \vec{W}^n_{R,m} \\ - |\bar{A}_{Roe}|_m (\Delta \vec{W}^n_{R,m} - \Delta \vec{W}^n_{L,m}) \Big\} . \tag{6.44}$$

In contrast to the Steger-Warming flux-vector splitting scheme, the above approximate linearisation (6.44) degrades the performance of the implicit scheme only slightly [29].

Viscous Flows

For the Navier-Stokes equations, we have to account also for the viscous fluxes in the implicit operator. The derivative $\partial \vec{F}_v / \partial \vec{W}$, i.e., the viscous flux Jacobian in Eq. (6.30) is in general not straightforward to obtain. Additional complexity arises due to the fact that the viscous flux vector contains derivatives of flow variables. For this reason, we either have to evaluate the viscous flux Jacobian by finite differences (Eq. (6.41) or we have to use a simplified formulation.

In the case of the TSL approximation of the Navier-Stokes equations (cf. Subsection 2.4.3 and Section A.5), it is possible to find the viscous flux Jacobian analytically by assuming locally constant dynamic viscosity and thermal conductivity coefficients. Then, according to the discussion in Appendix A.8, the term related to the viscous fluxes in Eq. (6.29) becomes

$$\frac{\partial (\vec{F}_v \Delta S)_m}{\partial \vec{W}} \Delta \vec{W}^n \approx \Big[(\bar{A}_v^*)_{R,m} \Delta \vec{W}^n_{R,m} - (\bar{A}_v^*)_{L,m} \Delta \vec{W}^n_{L,m} \Big] \Delta S_m \, . \tag{6.45}$$

In above Eq. (6.45), $\bar{A}_v^*$ stands for the viscous flux Jacobian given by Eq. (A.50) or Eq. (A.54) but without the spatial operators $\partial_\psi()$ (cf. Eqs. (A.53) and (A.58)). The Jacobians are evaluated using either the left or the right state for all variables except for the dynamic viscosity, which is determined from an arithmetic average. It should be mentioned that Eq. (6.45) leads to a second-order central difference approximation in the implicit operator, supposed the left and right state are computed with first-order accuracy.

6.2.3 ADI Scheme

The *Alternating Direction Implicit* (ADI) scheme was one of the first iterative implicit schemes [35]. The ADI scheme can be implemented on structured grids only. It is based on an approximate splitting (or, in other words, factorisation) of the implicit operator in Eq. (6.28) into two (in 2D) or three (in 3D) factors. Each factor contains the linearisation of the convective and viscous fluxes for one particular direction in the computational space. In 3D, this leads to the

formulation [35], [36]

$$
\begin{aligned}
&\left\{\frac{\Omega}{\Delta t}\bar{I} + \frac{\partial[(\vec{F}_c^I - \vec{F}_v^I)\Delta S^I]_{i+1/2}}{\partial \vec{W}} + \frac{\partial[(\vec{F}_c^I - \vec{F}_v^I)\Delta S^I]_{i-1/2}}{\partial \vec{W}}\right\} \\
&\left\{\frac{\Omega}{\Delta t}\bar{I} + \frac{\partial[(\vec{F}_c^J - \vec{F}_v^J)\Delta S^J]_{j+1/2}}{\partial \vec{W}} + \frac{\partial[(\vec{F}_c^J - \vec{F}_v^J)\Delta S^J]_{j-1/2}}{\partial \vec{W}}\right\} \\
&\left\{\frac{\Omega}{\Delta t}\bar{I} + \frac{\partial[(\vec{F}_c^K - \vec{F}_v^K)\Delta S^K]_{k+1/2}}{\partial \vec{W}} + \frac{\partial[(\vec{F}_c^K - \vec{F}_v^K)\Delta S^K]_{k-1/2}}{\partial \vec{W}}\right. \\
&\qquad\qquad\qquad\qquad\left. - \frac{\partial(\Omega\vec{Q})}{\partial \vec{W}}\right\}\Delta\vec{W}^n = -\vec{R}^n .
\end{aligned}
\tag{6.46}
$$

For clarity, the node indices i, j, k were omitted from Eq. (6.46) where not required. Furthermore, the superscripts I, J, K in Eq. (6.46) mark the flux vector or of the face area associated with certain coordinate in the computational space. For example, $\Delta S^I_{i+1/2,j,k}$ corresponds to ΔS_2 in Fig. 4.1b. By replacing the node indices by cell indices, the scheme in Eq. (6.46) can be applied to a cell-centred discretisation.

The derivatives of the convective and viscous fluxes in Eq. (6.46) can be evaluated as discussed in the previous subsection. The ADI method is traditionally coupled to the central scheme with artificial dissipation. In such a case, a formulation similar to that in Eq. (6.34) holds for each factor (of course, the linearisation of the source term is included in one factor only). In order to obtain a robust and efficient scheme, it is necessary to include a linearisation of the artificial dissipation terms in the implicit operator [37]-[39]. The linearisation is in general simplified by treating the spectral radii and the dissipation coefficients as independent of $\vec{W}$. Hence, according to Eq. (4.48) the factor, e.g., in the I-direction becomes

$$
\begin{aligned}
&\left\{\frac{\Omega}{\Delta t}\bar{I} + \frac{\partial[(\vec{F}_c^I - \vec{F}_v^I)\Delta S^I]_{i+1/2}}{\partial \vec{W}} + \frac{\partial[(\vec{F}_c^I - \vec{F}_v^I)\Delta S^I]_{i-1/2}}{\partial \vec{W}}\right. \\
&\qquad\qquad\left. - \frac{\partial(\vec{D}^I_{IM}\Delta S^I)_{i+1/2}}{\partial \vec{W}} - \frac{\partial(\vec{D}^I_{IM}\Delta S^I)_{i-1/2}}{\partial \vec{W}}\right\}
\end{aligned}
$$

with the implicit artificial dissipation term JST scheme, (cf. Eq. (4.50))

$$
\begin{aligned}
\frac{\partial(\vec{D}^I_{IM})_{i+1/2}}{\partial \vec{W}}\Delta\vec{W}^n \approx \hat{\Lambda}^S_{i+1/2}\Big[&(\epsilon^{(2)}_{IM})_{i+1/2}(\Delta\vec{W}^n_{i+1} - \Delta\vec{W}^n_i) \\
&- (\epsilon^{(4)}_{IM})_{i+1/2}(\Delta\vec{W}^n_{i+2} - 3\Delta\vec{W}^n_{i+1} + 3\Delta\vec{W}^n_i - \Delta\vec{W}^n_{i-1})\Big] .
\end{aligned}
\tag{6.47}
$$

The implicit dissipation terms in other directions are defined in similar way. It is also possible to retain only the second-order differences in the implicit operator [38], [39]. This reduces the matrix for each of the factors from block-pentadiagonal to block-tridiagonal form (as sketched in Fig. 6.1). However, as discussed in [39] this restricts the stability of the scheme.

The inversion of the implicit operator in Eq. (6.46) proceeds in three steps (two steps in 2D), i.e.,

$$
\begin{aligned}
(\mathbf{D}^I + \mathbf{L}^I + \mathbf{U}^I)\Delta\vec{W}^{(1)} &= -\vec{R}^n \\
(\mathbf{D}^J + \mathbf{L}^J + \mathbf{U}^J)\Delta\vec{W}^{(2)} &= \Delta\vec{W}^{(1)} \\
(\mathbf{D}^K + \mathbf{L}^K + \mathbf{U}^K)\Delta\vec{W}^{n} &= \Delta\vec{W}^{(2)}\,,
\end{aligned}
\tag{6.48}
$$

where $\mathbf{D}$ represents the diagonal, $\mathbf{L}$ the lower-diagonal and $\mathbf{U}$ the upper-diagonal terms, respectively. Each step requires the inversion of a block-tridiagonal or a block-pentadiagonal matrix (if $\epsilon_{IM}^{(4)} > 0$ in Eq. (6.47)). This is done by a direct solution method. In order to reduce the numerical effort, Pulliam and Chaussee [40] suggested a diagonalised form of the ADI scheme. Herewith, the block matrices (composed of convective and viscous flux Jacobians) are transformed into diagonal matrices. Hence, only non-block tri- or pentadiagonal matrices have to be inverted, which results in significant savings of computational work and memory [38]. Although the diagonalisation is strictly valid only for Euler equations, it can be employed for viscous flows as well [38]. The linearisation of the viscous fluxes (e.g., like in Eq. (6.45)) is then either omitted in the implicit operator, or it can be approximated by the viscous eigenvalue (Eq. (6.19)).

The splitting of the implicit operator introduces what is called the *factorisation error*. It is the difference between the implicit operator of the base scheme in Eq. (6.28) and the factorised operator. In the case of the ADI scheme, this error term is scaled by the factor $(\Delta t)^N$, where N denotes the number of space dimensions. This term causes the ADI scheme to loose its unconditional stability in 3D [41]. However, the stability is improved if the fourth-order differences of the artificial viscosity are included in the implicit operator [39]. In fact, the ADI scheme was successfully used for the solution of various 3-D problems [38], [42]. An interesting implementation of the ADI scheme on multiblock grids, which treats the block boundaries implicitly, was presented by Rosenfeld et al. [43].

The time step Δt can be computed in the same way as presented in Subsection 6.1.4, using Eq. (6.14). The optimal CFL number lies between 20 and 50.

6.2.4 LU-SGS Scheme

The implicit *Lower-Upper Symmetric Gauss-Seidel* (LU-SGS) scheme, which is also called the *Lower-Upper Symmetric Successive Overrelaxation* (LU-SSOR) scheme, became widely-used because of its low numerical complexity and modest memory requirements, which are both comparable to an explicit multistage scheme. Furthermore, the LU-SGS scheme can be implemented easily on vector and parallel computers. It can also be used on structured as well as on unstructured grids.

The LU-SGS scheme has its origins in the work of Jameson and Turkel [44], who considered decompositions of the implicit operator into lower and upper diagonally dominant factors. The LU-SGS method itself was introduced by Yoon and Jameson [45]-[47] as a relaxation method for solving the unfactored implicit scheme in Eq. (6.28). It was further developed and applied to 3-D viscous flow fields by Rieger and Jameson [48]. Since then, various researchers applied the LU-SGS scheme to viscous flows on structured [49]-[55] and on unstructured grids [56]-[60]. The LU-SGS approach is used often for the simulation of chemically reacting flows [61]-[65].

The LU-SGS scheme employs a simplification of the first-order accurate flux-vector splitting approach due to Steger and Warming (see Subsection 6.2.2) for the linearisation of the convective fluxes in Eq. (6.29). The linearisation is always kept the same regardless of the discretisation of the explicit operator. The LU-SGS scheme is further based on the factorisation of the implicit operator in Eq. (6.28) into the following three parts

$$(\mathbf{D}+\mathbf{L})\mathbf{D}^{-1}(\mathbf{D}+\mathbf{U})\Delta\vec{W}^n = -\vec{R}_I^n\,. \tag{6.49}$$

The factors are constructed such that $\mathbf{L}$ consists only of terms in the strictly lower triangular matrix, $\mathbf{U}$ of terms in the strictly upper triangular matrix and $\mathbf{D}$ of diagonal terms. It is important to remark that the number of factors remains still the same independent of the number of space dimensions.

The system matrix of the LU-SGS scheme (Eq. (6.49)) can be inverted in two steps – a forward and a backward sweep, i.e.,

$$\begin{aligned}(\mathbf{D}+\mathbf{L})\Delta\vec{W}^{(1)} &= -\vec{R}_I^n\\ (\mathbf{D}+\mathbf{U})\Delta\vec{W}^n &= \mathbf{D}\,\Delta\vec{W}_I^{(1)}\end{aligned} \tag{6.50}$$

with $\vec{W}^{n+1} = \vec{W}^n + \Delta\vec{W}^n$. The operators $\mathbf{L}$, $\mathbf{D}$, and $\mathbf{U}$ and also the inversion procedure differ on structured and unstructured grids, in some respects. Therefore, we shall discuss below each case separately.

LU-SGS on Structured Grids

On structured grids, the operators are defined as (see [45]-[47], and [62], [50])

$$
\begin{aligned}
\mathbf{L} &= \left(\bar{A}^+ + \bar{A}_v\right)_{i-1} \Delta S^I_{i-1/2} + \left(\bar{A}^+ + \bar{A}_v\right)_{j-1} \Delta S^J_{j-1/2} \\
&\quad + \left(\bar{A}^+ + \bar{A}_v\right)_{k-1} \Delta S^K_{k-1/2} \\
\mathbf{U} &= \left(\bar{A}^- - \bar{A}_v\right)_{i+1} \Delta S^I_{i+1/2} + \left(\bar{A}^- - \bar{A}_v\right)_{j+1} \Delta S^J_{j+1/2} \\
&\quad + \left(\bar{A}^- - \bar{A}_v\right)_{k+1} \Delta S^K_{k+1/2} \qquad (6.51) \\
\mathbf{D} &= \frac{\Omega}{\Delta t}\bar{I} + \left(\bar{A}^- - \bar{A}_v\right) \Delta S^I_{i-1/2} + \left(\bar{A}^- - \bar{A}_v\right) \Delta S^J_{j-1/2} \\
&\quad + \left(\bar{A}^- - \bar{A}_v\right) \Delta S^K_{k-1/2} + \left(\bar{A}^+ + \bar{A}_v\right) \Delta S^I_{i+1/2} \\
&\quad + \left(\bar{A}^+ + \bar{A}_v\right) \Delta S^J_{j+1/2} + \left(\bar{A}^+ + \bar{A}_v\right) \Delta S^K_{k+1/2} - \frac{\partial(\Omega\vec{Q})}{\partial\vec{W}}\,.
\end{aligned}
$$

For better readability, only those node indices (or cell indices in the case of a cell-centred scheme) are shown in Eq. (6.51), which differ from i, j, k. The superscripts i, j, k at ΔS indicate the direction in the computational space. The unit normal vectors in the positive/negative flux Jacobians $\bar{A}^\pm$ and in the viscous flux Jacobians $\bar{A}_v$ are evaluated at the same side of the control volume like the associated face areas ΔS. Note that the unit normal vectors are assumed to point outwards of the control volume. In contrast, in various references it is supposed that the unit normal vectors from opposite sides of the control volume point in the same direction.

The viscous flux Jacobians in Eq. (6.51) are either computed numerically, or are replaced by their TSL approximation, corresponding to Eq. (6.45). It is possible to apply the TSL approximation in all computational coordinates, regardless of the actual orientation of the boundary layer(s). A further simplification consists of substituting the viscous flux Jacobians by the viscous spectral radii (Eq. (6.19)), i.e., $\bar{A}_v \Delta S \approx \hat{\Lambda}_v$, as suggested in [57].

The split convective flux Jacobians $\bar{A}^\pm$ are constructed in such a way that the eigenvalues of the (+) matrices are all non-negative, and of the (−) matrices are all non-positive. In general, the matrices are defined as [44]

$$
\bar{A}^\pm \Delta S = \frac{1}{2}\left(\bar{A}_c \Delta S \pm r_A \bar{I}\right)\,, \quad r_A = \omega\,\hat{\Lambda}_c\,, \qquad (6.52)
$$

where $\bar{A}_c$ stands for the convective flux Jacobian (Section A.7) and $\hat{\Lambda}_c$ represents the spectral radius of the convective flux Jacobian (given by Eq. (4.53) or Eq. (6.15)), respectively. Note the similarity between the above approximation

(6.52) and Eq. (6.40), when the derivatives of $\bar{A}_c$ are neglected. The factor ω in Eq. (6.52) represents an overrelaxation parameter. It also determines the amount of implicit dissipation and hence influences the convergence properties of the scheme. The factor can be chosen in the range $1 < \omega \leq 2$. Higher values of ω increase the stability of the LU-SGS scheme, but may slow down the convergence to steady state. The definition of the Jacobians $\bar{A}^{\pm}$ in Eq. (6.52) ensures a diagonally dominant system matrix, which is very important for the efficiency and robustness of the iterative inversion procedure (6.50).

The splitting according to Eq. (6.52) together with averaged face vectors allow a simplified evaluation of the diagonal operator **D**

$$\mathbf{D} = \left[\frac{\Omega}{\Delta t} + \omega\left(\hat{\Lambda}_c^I + \hat{\Lambda}_c^J + \hat{\Lambda}_c^K\right)\right]\bar{I} + 2\left(\bar{A}_v^I \Delta S^I + \bar{A}_v^J \Delta S^J + \bar{A}_v^K \Delta S^K\right) - \frac{\partial(\Omega\vec{Q})}{\partial\vec{W}} . \tag{6.53}$$

The spectral radii of the convective flux Jacobians $\hat{\Lambda}_c$ are given in Eq. (6.15). The face areas and normal vectors are averaged in the respective I-, J-, or K-direction according to Eq. (6.16). As we shall see immediately, this approximation helps to reduce the operation count and memory requirements significantly.

A distinguishing feature of the LU-SGS method is how the forward and the backward sweep in Eq. (6.50) are carried out. In 2D, the sweeps are accomplished along diagonal lines $(i + j)$ = const. in computational space. This is depicted in Fig. 6.5 for the forward sweep (first line of Eq. (6.50)). In this way, the off-diagonal terms involved in the **L** and the **U** operator become known from the previous part of a sweep (denoted by crosses in Fig. 6.5). In 3D, the implicit operator is inverted on $i + j + k$ = const. planes, as sketched in Fig. 6.6. Hence, the LU-SGS scheme can be written as

$$\begin{aligned} \mathbf{D}\,\Delta\vec{W}^{(1)}_{i,j,k} &= -\vec{R}^n_{i,j,k} - \mathbf{L}\,\Delta\vec{W}^{(1)} \\ \mathbf{D}\,\Delta\vec{W}^{n}_{i,j,k} &= \mathbf{D}\,\Delta\vec{W}^{(1)}_{i,j,k} - \mathbf{U}\,\Delta\vec{W}^{n} . \end{aligned} \tag{6.54}$$

As we can see from Eq. (6.54), the only term which needs to be inverted is the diagonal term **D**. Thus, the LU-SGS methodology transforms the inversion of a sparse banded matrix into the inversion of a block-diagonal matrix. Furthermore, if the viscous flux Jacobians in Eq. (6.53) are approximated by the viscous spectral radii, the operator **D** becomes a purely diagonal matrix. Hence, the LU-SGS scheme requires little computational effort compared to other implicit schemes (e.g., the ADI scheme discussed previously). Furthermore, the inversion of the diagonal operator can be carried out independently for each node (cell) of the diagonal plane, which makes the scheme easy to vectorise. The indices of the nodes/cells on the diagonal planes can be obtained with the following pseudo-code [66]:

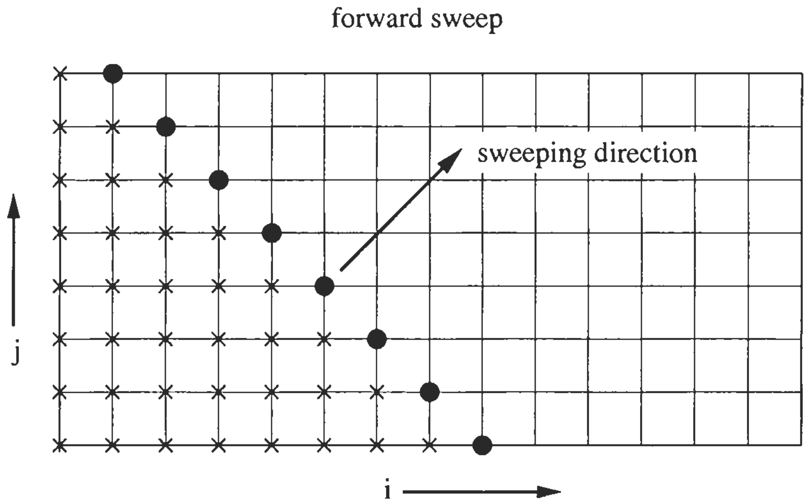


Figure 6.5: Sweeping direction of the LU-SGS scheme in computational space: • denotes where the operator **D** is currently inverted (line $i + j$ = const.); × denotes the already updated values of **L**.

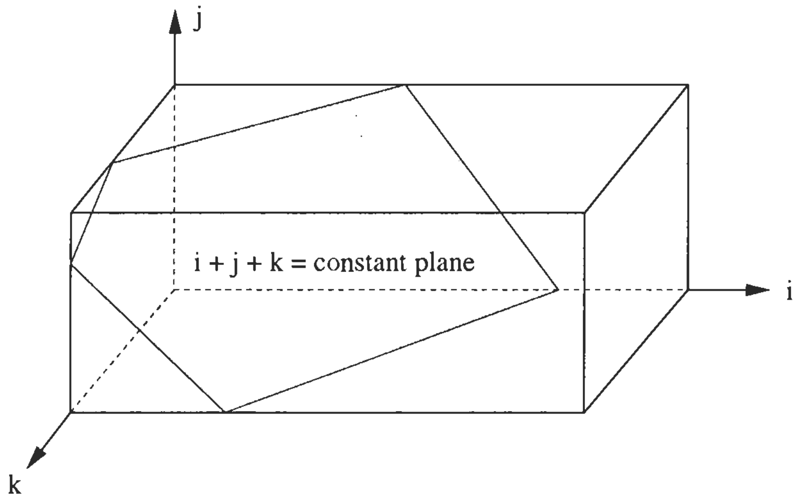


Figure 6.6: Diagonal plane of sweep in computational space for the implicit LU-SGS scheme in 3D.

```
DO plane = 1, nplanes
   DO k = 1, kmax
      DO j = 1, jmax
         DO i = 1, imax
            IF (i+j+k = plane+2) store indices
         ENDDO
      ENDDO
   ENDDO
ENDDO
```

The number of diagonal planes is: $nplanes = imax + jmax + kmax - 2$. Obviously, the above code can be optimised for higher computational efficiency.

In order to avoid explicit evaluation and storage of the convective flux Jacobians in **L** and **U**, the products $\bar{A}^{\pm}\Delta\vec{W}^n$ can be substituted by Taylor series expansion of the fluxes [48]. Using Eq. (6.52), we can write

$$(\bar{A}^{\pm}\Delta S)\,\Delta\vec{W}^n \approx \frac{1}{2}\left(\Delta\vec{F}_c\Delta S \pm r_A \bar{I}\Delta\vec{W}^n\right) \tag{6.55}$$

with the update of the convective fluxes

$$\Delta\vec{F}_c = \vec{F}_c^{n+1} - \vec{F}_c^{n}\,. \tag{6.56}$$

The simplification given by Eq. (6.55) is possible due to the sweeping along diagonal planes, since $\vec{F}_c^{n+1}$ is then known. This leads to a further significant decrease of the numerical effort of the LU-SGS scheme.

The time step Δt can be computed in the same way as presented in Subsection 6.1.4, using Eq. (6.14). However, it should be noted that the implicit LU-SGS scheme in Eq. (6.49) represents an approximate Newton iteration in the case of $\Delta t \to \infty$ as stated by Rieger and Jameson [48]. Thus in general, CFL number in the order of 10^4 to 10^6 are used in practice for stationary flows. The convergence is then controlled by the overrelaxation parameter ω. For the simulation of unsteady flows, we may employ the formulation presented below in Section 6.3. Another possibility is to use the modified version of the LU-SGS scheme described in Ref. [67].

LU-SGS on Unstructured Grids

Here, the operators read for a median-dual scheme [57]-[59]

$$\begin{aligned}
\mathbf{L} &= \sum_{j\in L(i)} \left[\bar{A}_j^{+} + (\bar{A}_v)_j\right]\Delta S_{ij} \\
\mathbf{U} &= \sum_{j\in U(i)} \left[\bar{A}_j^{-} - (\bar{A}_v)_j\right]\Delta S_{ij} \\
\mathbf{D} &= \frac{\Omega_i}{\Delta t_i}\bar{I} + \frac{\omega}{2}(\hat{\Lambda}_c)_i + \sum_{j=1}^{N_F}(\bar{A}_v)_i\Delta S_{ij} - \frac{\partial(\Omega_i\vec{Q}_i)}{\partial\vec{W}}\,.
\end{aligned} \tag{6.57}$$

In Eq. (6.57), $L(i)$, and $U(i)$ denote the nearest neighbours of node i which belong to the lower (upper) matrix, ΔS_{ij} represents the face area associated with the edge ij (see Fig. 5.9), and N_F stands for the number of faces of the control volume Ω_i, respectively. The spectral radius of the convective fluxes $(\hat{\Lambda}_c)_i$ is computed by Eq. (6.21). The viscous flux Jacobian $\bar{A}_v$ can be again approximated by its spectral radius [57]. In this case, the diagonal operator becomes

$$\mathbf{D} = \frac{\Omega_i}{\Delta t_i}\bar{I} + \frac{\omega}{2}(\hat{\Lambda}_c)_i + (\hat{\Lambda}_v)_i - \frac{\partial(\Omega_i \vec{Q}_i)}{\partial \vec{W}}, \tag{6.58}$$

where $(\hat{\Lambda}_v)_i$ is evaluated according to Eq. (6.21). Formulae similar to Eq. (6.57) and (6.58) can be obtained in the case of the cell-centred scheme. The major difference is that the summation in the **L** and the **U** operator is conducted over faces of the cell instead of incident edges.

The sets $L(i)$ and $U(i)$ in Eq. (6.57) should fulfil the same function as the diagonal planes on structured grids. For this reason, it is necessary to arrange the nodes (cells) into layers such that [57]:

- nodes i (cells I) of a current layer have connections to layers with previously updated flow variables – otherwise the LU-SGS scheme degenerates to a Jacobi iteration,
- nodes (cells) in a layer are not connected to each other – otherwise the scheme could not be vectorised.

The layers can be generated for the median-dual scheme with a procedure described in Ref. [57]. Approaches for the cell-centred scheme were suggested in [68], [69].

An appropriate definition of the sets $L(i)$ and $U(i)$ allows for the following two-step inversion procedure [57]-[59]

$$\begin{aligned}
\mathbf{D}\,\Delta\vec{W}_i^{(1)} &= -\vec{R}_i^n - \sum_{j\in L(i)} \frac{1}{2}\left[(\Delta F_c^{(1)})_j \Delta S_{ij} + (r_A^*)_j \bar{I}\,\Delta\vec{W}_j^{(1)}\right] \\
\mathbf{D}\,\Delta\vec{W}_i^n &= \mathbf{D}\,\Delta\vec{W}_i^{(1)} - \sum_{j\in U(i)} \frac{1}{2}\left[(\Delta F_c^n)_j \Delta S_{ij} - (r_A^*)_j \bar{I}\,\Delta\vec{W}_j^n\right],
\end{aligned} \tag{6.59}$$

where the viscous Jacobians were approximated by their spectral radii and where the positive/negative Jacobians were linearised according to Eq. (6.55). Furthermore, the factor $(r_A^*)_j$ is defined as

$$\begin{aligned}
(r_A^*)_j &= \omega\left(|\vec{v}_j \cdot \vec{n}_{ij}| + c_j\right)\Delta S_{ij} \\
&+ \frac{\Delta S_{i,j}}{||\vec{r}_j - \vec{r}_i||_2}\left[\max\left(\frac{4}{3\,\rho_j}, \frac{\gamma_j}{\rho_j}\right)\left(\frac{\mu_L}{Pr_L} + \frac{\mu_T}{Pr_T}\right)_j\right]
\end{aligned} \tag{6.60}$$

with $||\vec{r}_j - \vec{r}_i||_2$ being the length of the edge ij.

The time step can be computed in the same way as for the explicit scheme (Eq. (6.20)). However, the viscous eigenvalue should be omitted, since the viscous terms are already contained in the implicit operator and hence do not reduce the time step as in the case of an explicit scheme. The CFL number can be chosen in the range from 10^4 to 10^6 for steady flows. The convergence speed and the robustness of the LU-SGS scheme can then be tuned using the overrelaxation parameter ω.

6.2.5 Newton-Krylov Method

First of all, let us rewrite the implicit scheme given by Eq. (6.28) as

$$\bar{J}\,\Delta\vec{W}^n = -\vec{R}^n\,, \tag{6.61}$$

where $\bar{J}$ represents the implicit operator (system matrix). As we already saw, $\bar{J}$ constitutes a large, sparse, and generally non-symmetric matrix. In the previous subsections, we discussed two methods that decompose $\bar{J}$ into several factors which can be each more easily inverted than $\bar{J}$ itself. However, due to the factorisation error (and approximate linearisation of $\vec{R}^{n+1}$), only a linear convergence to steady state can be achieved. In order to obtain the quadratic convergence of Newton's method for the solution of non-linear equations, four conditions must be fulfilled:

- the linearisation of the residual must be exact,
- $\bar{J}$ must be accurately inverted,
- the time step has to be $\Delta t \rightarrow \infty$,
- initial solution must be, in some sense, close to the final solution.

Obviously, the main obstacles that have to be overcome are the linearisation and the inversion of the full system matrix.

A particularly suitable class of iterative techniques for the solution of large linear equation systems are the so-called *Krylov-subspace* methods. Several were proposed for the inversion of matrices which arise in CFD. Examples are the *Conjugate Gradient Squared* (CGS) method [70], the *Bi-Conjugate Gradient Stabilised* (Bi-CGSTAB) scheme [71], or the *Transpose-Free Quasi-Minimum Residual* (TFQMR) approach [72]. However, the most successful Krylov subspace method became the *Generalised Minimal Residual* (GMRES) technique, which was originally suggested by Saad and Schulz [73], [74]. Since then, the GMRES method was improved and augmented by several researchers. [75]-[78]. Because of its popularity, we shall focus on the GMRES approach in the following. Nevertheless, most of what we shall discuss also applies to the other Krylov subspace methods.

GMRES Method

As we mentioned in Subsection 3.2.2 (see also Appendix A.10), the GMRES method minimises the norm of the global residual, i.e., $\|\bar{J}\,\Delta\vec{W}^n + \vec{R}^n\|$ over a set of m orthonormal vectors (search directions), which span the Krylov subspace $\mathcal{K}_m$ given by Eq. (3.10). The GMRES algorithm can be summarised as follows:

1. guess a starting solution $\vec{W}_0^n$ and evaluate the initial residual vector $\vec{r}_0 = \bar{J}\,\Delta\vec{W}_0^n + \vec{R}^n$,
2. generate the m search directions (by Gram-Schmidt orthogonalisation),
3. solve the minimisation problem,
4. form an approximate solution of Eq. (6.61) as $\Delta\vec{W}^n = \Delta\vec{W}_0^n + \vec{y}_m$.

Since the memory requirements increase linearly with the number of search directions, m is restricted to values between 10 and 40 in practice. This might not be sufficient for a converged solution $\Delta\vec{W}^n$. Thus, the GMRES method has to be restarted, i.e., we set $\Delta\vec{W}_0^n = \Delta\vec{W}^n$, compute $\vec{r}_0$ and proceed with step 2. As pointed out in Ref. [79], instead of working with a constant number of search directions, m should be reduced if the norm of the global residual drops below a specified tolerance. In this way, a large number of operations can be saved in later stages of the Newton iteration.

Computation of Flux Jacobian

GMRES and other Krylov subspace methods allow us to circumvent an explicit computation and storage of the flux Jacobian $\partial\vec{R}/\partial\vec{W}$. The idea is based on the observation that the methods rely only on matrix-vector products of the form $\bar{J}\,\Delta\vec{W}^n$ and do not need the matrix $\bar{J}$ explicitly. The product of the flux Jacobian with the solution update can be approximated by a simple finite difference as

$$\frac{\partial\vec{R}}{\partial\vec{W}}\,\Delta\vec{W}^n \approx \frac{\vec{R}(\vec{W}^n + h\,\Delta\vec{W}^n) - \vec{R}(\vec{W}^n)}{h} \tag{6.62}$$

which requires only two evaluations of the residual. The stepsize h has to be chosen with some care, in order to minimise the numerical error [31]. One particularly suitable formulation reads [80]

$$h = \frac{\sqrt{\epsilon}}{\|\Delta\vec{W}^n\|_2^2}\max\left\{|d|,\ \mathrm{typ}\left[\vec{W}^n\cdot|\Delta\vec{W}^n|\right]\right\}\,\mathrm{sign}(d)\,, \tag{6.63}$$

where ϵ denotes the machine accuracy, d the scalar product $\vec{W}^n\cdot\Delta\vec{W}^n$, $|\Delta\vec{W}^n|$ is the vector $\Delta\vec{W}^n$ with all elements set to their absolute values, and finally $\mathrm{typ}\,U$ represents a typical size of U. Appart from saving memory and operations, there is an even more important advantage of the finite-difference approximation. Namely, numerically accurate linearisation of a high-order residual $\vec{R}^n$ (including boundary conditions, limiters, source terms, etc.) can easily be

achieved. Thus, the quadratic convergence of Newton's scheme can be realised at moderate costs. For this reason, we speak of such a scheme as of *Newton-Krylov* approach [80]-[82].

Preconditioning

The efficiency of Krylov-subspace methods depends strongly on a good preconditioner. Its purpose is to cluster the eigenvalues of the system matrix $\bar{J}$ around unity. Thus, instead of Equation (6.61), the left- or right-preconditioned system according to Eq. (3.11) is solved. Using Eqs. (6.61) and (6.62) together with the condition $\Delta t \rightarrow \infty$, the Newton-Krylov method becomes

$$\bar{P}_L \, \frac{\vec{R}(\vec{W}^n + h\,\Delta\vec{W}^n) - \vec{R}(\vec{W}^n)}{h} = -\bar{P}_L\,\vec{R}^n \tag{6.64}$$

with left preconditioning, and

$$\frac{\vec{R}(\vec{W}^n + h\bar{P}_R\,\Delta\vec{W}^*) - \vec{R}(\vec{W}^n)}{h} = -\vec{R}^n\,, \quad \bar{P}_R^{-1}\Delta\vec{W}^n = \Delta\vec{W}^* \tag{6.65}$$

in the case of right preconditioning, respectively. The main difference between the two preconditioning methodologies is that left preconditioning scales the residual $\vec{R}^n$ whereas right preconditioning does not. This has to be kept in mind when the convergence of the Krylov method is monitored.

Obviously, the preconditioner should be as close as possible to the inverse of the system matrix ($P \approx \bar{J}^{-1}$). But on the other hand, it should be invertible with low numerical effort. Therefore, we have to find an optimal tradeoff between the convergence speed of the Krylov method and the time spend for inverting the preconditioning matrix. One of the most successful preconditioners is the *Incomplete Lower Upper* factorisation method [83], [84] with varying level of fill-in (mostly zero, designated as ILU(0)). The ILU preconditioner is especially efficient in the case of viscous, turbulent flows, i.e., for stiff equations [85]. In order to obtain a good performance on unstructured grids, it is necessary to reorder the elements of the system matrix such that the bandwidth is reduced. We discussed the RCM renumbering strategy [22], [23] in Section 6.2.1 already.

A serious disadvantage of the ILU preconditioning scheme is that elements of the matrix $\bar{J}$ have to be computed (see Subsection 6.2.2) and stored. Therefore, some authors suggested to employ the LU-SGS scheme as a preconditioner [86], [87]. Hence, when Eq. (6.62) is applied, the formation and storage of $\bar{J}$ is completely avoided. However, it was demonstrated in [85] on behalf of several 2-D cases that the GMRES method with LU-SGS is inferior to GMRES combined with ILU(0) in terms of CPU-time. Nevertheless, the LU-SGS scheme (possibly coupled with multigrid) still represents an attractive alternative, particularly in 3D.

Start-Up Problem

The time step of the implicit Newton-Krylov method is infinitely large. However, it is advisable to use small time steps at the beginning of the Newton iteration process. The reason is that the flow solution is in general far from the steady state at the beginning of the solution process, i.e., the root of the nonlinear equation

$$\vec{R}(\vec{W}) = 0\,,$$

and this may cause a breakdown of the Newton iteration. One possible remedy is the so-called ***Switched Evolution Relaxation*** (SER) technique [88]. Here, the term $\Omega/\Delta t$ is retained in Eq. (6.61). The time step is evaluated in the same way as presented for the explicit scheme (Eq. (6.14) or Eq. (6.20) without the viscous eigenvalue). The CFL number σ is increased starting from a small initial value correspondingly to the reduction of the 2-norm of the residual, i.e.,

$$\sigma^{n+1} = \sigma^n \frac{\|\vec{R}^{n-1}\|_2}{\|\vec{R}^n\|_2} \tag{6.66}$$

Hence, the convergence of the iteration procedure (6.61) will be at first linear, but it approaches the quadratic convergence of Newton's method for large CFL numbers. A further effect of the time term is the increased diagonal dominance of $\bar{J}$ (inverse proportional to $\Omega/\Delta t$), which will help to stabilise the iteration. In Ref. [82], it was suggested to clip σ^{n+1} in Eq. (6.66) such that it increases by maximum factor of two and decreases less than factor of ten.

A further approach which can be used to overcome the start-up problems of Newton's method consist of grid sequencing, where the initial solution is obtained on a sequence of coarser grids and interpolated onto finer grids. Another possibility is to use a numerically cheap but robust iteration scheme for the initial guess. For example, we could start with a multigrid scheme driven by the LU-SGS method and then switch to GMRES with LU-SGS as preconditioner. This might be particularly interesting for viscous turbulent flows, where the convergence of a multigrid scheme usually slows down after the initial phase. However, the global flow solution is then already close to the steady state.

6.3 Methodologies for Unsteady Flows

The simulation of unsteady flow phenomena is becoming increasingly important in many engineering disciplines. Examples are the interaction between stationary and rotating parts in turbomachinery, piston engines, fluid-structure interaction, helicopter aerodynamics, aeroacoustics, DNS or LES of turbulent flows, detonations, etc. Clearly, the simulation has to be conducted efficiently and with accuracy adequate for the problem being solved.

Explicit schemes represent the best choice for certain unsteady applications when the time scales are comparable to the spatial scales over the eigenvalue, i.e., when the CFL number dictated by the physics is of the order of unity. This is for example the case in aeroacoustics, DNS, and LES. For such applications, explicit Runge-Kutta schemes are quite popular. Since the global physical phenomena evolve much slower than the solution changes locally in these applications, it is necessary to integrate over a long period of physical time. In order to do this accurately, the temporal resolution of the explicit scheme has to be of 3rd or higher order. This requires the use of Runge-Kutta methods different to that we presented in Section 6.1 (see, e.g., [89], [90]).

In other cases, where the physical time scales are large in comparison to the spatial scales divided by the eigenvalue (e.g., flutter, rotor-stator interaction, etc.), the CFL number can be chosen in the order of several hundreds or even thousands without impairing the accuracy of the simulation. Obviously, in such cases an implicit scheme is more appropriate. In the following, we shall discuss a particular technique known as the *dual time-stepping* approach, which is very often employed for unsteady flows.

The dual time-stepping approach is based on the second-order time accurate version of the basic non-linear scheme in Eq. (6.2). For this purpose we set $\beta = 1$ and $\omega = 1/2$ in Eq. (6.2). Hence, we obtain

$$\frac{3(\Omega\bar{M})_I^{n+1}\vec{W}_I^{n+1} - 4(\Omega\bar{M})_I^n\vec{W}_I^n + (\Omega\bar{M})_I^{n-1}\vec{W}_I^{n-1}}{2\,\Delta t} = -\vec{R}_I^{n+1}\,, \tag{6.67}$$

where Δt denotes the **global** physical time step and $\bar{M}$ the mass matrix, respectively. Equation (6.67) constitutes a 3-point backward-difference approximation of the time derivative in Eq. (6.1). In order to solve the system of non-linear equations given by Eq. (6.67), we can use either Newton's method or a time-stepping methodology. The latter can be written as

$$\frac{\partial}{\partial t^*}\left(\Omega_I^{n+1}\vec{W}_I^*\right) = -\vec{R}_I^*(\vec{W}^*)\,, \tag{6.68}$$

where $\vec{W}^*$ is the approximation to $\vec{W}^{n+1}$ and t^* denotes a pseudo-time variable. Note that there is no mass matrix in the time derivative. The **unsteady** residual is defined as

$$\vec{R}_I^*(\vec{W}^*) = \vec{R}_I(\vec{W}^*) + \frac{3}{2\,\Delta t}(\Omega\bar{M})_I^{n+1}\vec{W}_I^* - \vec{Q}_I^*\,. \tag{6.69}$$

All terms which are constant during the time-stepping in Eq. (6.68) are gathered in a source term, i.e.,

$$\vec{Q}_I^* = \frac{2}{\Delta t}(\Omega \bar{M})_I^n \vec{W}_I^n - \frac{1}{2\,\Delta t}(\Omega \bar{M})_I^{n-1} \vec{W}_I^{n-1} \,. \tag{6.70}$$

In the case of moving and/or deforming grids, the new size of the control volume, i.e., Ω^{n+1} in Eq. (6.68) has to satisfy the Geometry Conservation Law (see Appendix A.4 for details and references).

The stationary solution of Eq. (6.68) corresponds to the flow variables at the new time level, i.e., $\vec{W}^* = \vec{W}^{n+1}$. Since $\vec{R}_I^* = 0$ at steady state in pseudo time, Equation (6.67) is fulfilled. Any of the previously presented explicit or implicit time-marching schemes can be employed for the solution of the system of equations (6.68) in pseudo time. In the following, we shall discuss the implementation of the dual time-stepping approach for explicit multistage and implicit schemes.

6.3.1 Dual Time-Stepping for Explicit Multistage Schemes

Jameson [91] first implemented the dual-time methodology using an explicit multistage scheme accelerated by local time-stepping and multigrid. The significant advantage of this approach is that the physical time step is not restricted as usual in explicit methods. It can be chosen based solely on the flow physics. On the other hand, additional storage is needed only for the source term $\vec{Q}^*$, which makes the approach very attractive. An m-stage explicit scheme for the solution of the pseudo-time problem (6.68) reads

$$\begin{aligned}
\vec{W}_I^{(0)} &= (\vec{W}_I^*)^l \\
\vec{W}_I^{(1)} &= \vec{W}_I^{(0)} - \frac{\alpha_1 \Delta t_I^*}{\Omega_I^{n+1}} \vec{R}_I^*(\vec{W}^{(0)}) \\
\vec{W}_I^{(2)} &= \vec{W}_I^{(0)} - \frac{\alpha_2 \Delta t_I^*}{\Omega_I^{n+1}} \vec{R}_I^*(\vec{W}^{(1)}) \\
&\vdots \\
(\vec{W}_I^*)^{l+1} &= \vec{W}_I^{(0)} - \frac{\alpha_m \Delta t_I^*}{\Omega_I^{n+1}} \vec{R}_I^*(\vec{W}^{(m-1)}) \,,
\end{aligned} \tag{6.71}$$

where l denotes the actual and $(l+1)$ the new pseudo-time level, respectively. The time-marching process is started either with $(\vec{W}^*)^l = \vec{W}^n$ or with a value extrapolated from previous physical time steps, e.g., [92]

$$(\vec{W}_I^*)^l = \vec{W}^n + \frac{3\vec{W}^n - 4\vec{W}^{n-1} + \vec{W}^{n-2}}{2} \,. \tag{6.72}$$

It is continued until $(\vec{W}_I^*)^{l+1}$ approximates $\vec{W}_I^{n+1}$ with sufficient accuracy (usually when the residual $\vec{R}_I^*$ was reduced by two or three orders of magnitude).

After that, the next **physical** time step is conducted. The pseudo time step Δt^* is computed in the same way that we saw in Subsection 6.1.4.

Arnone et al. [92] pointed out that the multistage scheme (6.71) becomes unstable when the physical time step Δt is of the order of the pseudo time step Δt^* or smaller. Melson et al. [93] demonstrated that the instability is caused by the term

$$\frac{3}{2\,\Delta t}(\Omega\bar{M})_I^{n+1}\vec{W}_I^*$$

in Eq. (6.69), which becomes significant for small Δt. They suggested an implicit treatment of this term. Thus, we have to modify the multistage scheme in Eq. (6.71) such that the k-th stage becomes [93]

$$\begin{aligned}\vec{W}_I^{(k)} = \vec{W}_I^{(0)} - \frac{\alpha_k \Delta t_I^*}{\Omega_I^{n+1}} \left[\bar{I} + \frac{3}{2\,\Delta t}\alpha_k \Delta t_I^* \bar{M}^{n+1}\right]^{-1} \\ \cdot \left[\vec{R}_I(\vec{W}^{(k-1)}) - \vec{Q}_I^*\right] .\end{aligned} \tag{6.73}$$

The same methodology can also be applied to a hybrid multistage scheme (see Subsection 6.1.2). The above formulation (6.73) is stable for any physical time step Δt [93].

In the case of cell-centred schemes, the mass matrix $\bar{M}^{n+1}$ in Eq. (6.73) can be lumped (substituted by the identity matrix) without reducing the solution accuracy. In this way, the term

$$\left[\bar{I} + \frac{3}{2\,\Delta t}\alpha_k \Delta t_I^* \bar{M}^{n+1}\right]^{-1}$$

in Eq. (6.73 is turned into a scalar value. However, we have to account for the mass matrix in the case of a cell-vertex spatial discretisation schemes. Otherwise, the multistage scheme will be unstable for small physical time steps. In order to circumvent the expensive inversion of $\bar{M}^{n+1}$, Venkatakrishnan [94] and Venkatakrishnan and Mavriplis [95] suggested the following modification of Eq. (6.73)

$$\begin{aligned}\vec{W}_I^{(k)} = \vec{W}_I^{(0)} - \frac{\alpha_k \Delta t_I^*}{\Omega_I^{n+1}} \left[1 + \frac{3}{2\,\Delta t}\alpha_k \Delta t_I^* \beta\right]^{-1} \\ \cdot \left[\vec{R}_I^*(\vec{W}^{(k-1)}) - \frac{3}{2\,\Delta t}\Omega_I^{n+1}\beta\,\vec{W}_I^{(k-1)}\right] .\end{aligned} \tag{6.74}$$

The parameter β can now be utilised to stabilise the time-stepping scheme. In practice, setting $\beta = 2$ was found sufficient [94], [95].

The dual time-stepping approach, where the solution in pseudo time is obtained by an explicit multistage scheme, is widely used. The highest computational efficiency results when the multistage scheme is accelerated by local time-stepping (in t^*) and multigrid. The reason is that the multigrid scheme converges quickly to the stationary solution of Eq. (6.68). Examples of applications on structured grids can be found in Refs. [91]-[93] as well as in [96]-[98].

Implementations of the methodology on unstructured grids were described, e.g., in [94], [95] and [99].

6.3.2 Dual Time-Stepping for Implicit Schemes

The implementation of an implicit scheme for the solution of Eq. (6.68) in pseudo time t^* proceeds in the same way as outlined in Section 6.2. First of all, we formulate Eq. (6.68) as an nonlinear implicit scheme, i.e.,

$$\frac{\partial}{\partial t^*}\left(\Omega_I^{n+1}\vec{W}_I^*\right) = -(\vec{R}_I^*)^{l+1} \tag{6.75}$$

with $(l+1)$ being the new pseudo-time level. Note again the absence of $\bar{M}$ in time derivative. The unsteady residual, which is defined in Eq. (6.69), can be linearised in pseudo time as follows

$$(\vec{R}^*)^{l+1} \approx (\vec{R}^*)^l + \frac{\partial \vec{R}^*}{\partial \vec{W}^*}\Delta\vec{W}^*\,, \tag{6.76}$$

where $\Delta\vec{W}^* = (\vec{W}^*)^{l+1} - (\vec{W}^*)^l$ and the flux Jacobian is defined as

$$\frac{\partial \vec{R}^*}{\partial \vec{W}^*} = \frac{\partial \vec{R}}{\partial \vec{W}} + \frac{3}{2\,\Delta t}(\Omega\bar{M})^{n+1}\,. \tag{6.77}$$

If we insert the above linearisation into Eq. (6.75), we obtain the unfactored implicit scheme [100]

$$\left[\left(\frac{1}{\Delta t_I^*} + \frac{3}{2\,\Delta t}\right)(\Omega\bar{M})_I^{n+1} + \left(\frac{\partial \vec{R}}{\partial \vec{W}}\right)_I\right]\Delta\vec{W}^* = -(\vec{R}_I^*)^l\,. \tag{6.78}$$

Any of the methodologies presented in Section 6.2 can be employed for the solution of the system (6.78). A detailed discussion of time-accurate implicit methods can be found in [101]. For recent examples of implementations, the reader is referred to, e.g., [100] and [102].

Bibliography

[1] Jameson, A.; Schmidt, W.; Turkel, E.: *Numerical Solutions of the Euler Equations by Finite Volume Methods Using Runge-Kutta Time-Stepping Schemes.* AIAA Paper 81-1259, 1981.

[2] Van Leer, B.; Tai, C.-H.; Powell, K.G.: *Design of Optimally Smoothing Multi-Stage Schemes for the Euler Equations.* AIAA Paper 89-1933, 1989.

[3] Tai, C.-H.; Sheu, J.-H.; van Leer, B.: *Optimal Multistage Schemes for Euler Equations with Residual Smoothing.* AIAA Journal, 33 (1995), pp. 1008-1016.

[4] Tai, C.-H.; Sheu, J.-H.; Tzeng, P.-Y.: *Improvement of Explicit Multistage Schemes for Central Spatial Discretization.* AIAA Journal, 34 (1996), pp. 185-188.

[5] Martinelli, L.: *Calculations of Viscous Flows with a Multigrid Method.* PhD Thesis, Dept. of Mechanical and Aerospace Engineering, Princeton University, 1987.

[6] Mavriplis, D.J.; Jameson, A.: *Multigrid Solution of the Navier-Stokes Equations on Triangular Meshes.* AIAA Journal, 28 (1990), pp. 1415-1425.

[7] Lafon, A.; Yee, H.C.: *On the Numerical Treatment of Nonlinear Source Terms in Reaction-Convection Equations.* AIAA Paper 92-0419, 1992.

[8] Curtiss, C.F.; Hirschfelder, J.O.: *Integration of Stiff Equations.* Proc. National Academy of Sciences of the USA, Vol. 38, 1952.

[9] Bussing, T.R.A.: *A Finite Volume Method for the Navier-Stokes Equations with Finite Rate Chemistry.* PhD Thesis, Dept. of Aeronautics and Astronautics, Massachusetts Inst. of Technology, 1985.

[10] Bussing, T.R.A.; Murman, E.M.: *Finite-Volume Method for the Calculation of Compressible Chemically Reacting Flows.* AIAA Journal, 26 (1988), pp. 1070-1078.

[11] Kunz, R.F.; Lakshminarayana, B.: *Stability of Explicit Navier-Stokes Procedures Using k-ε and k-ε/Algebraic Reynolds Stress Turbulence Models.* J. Computational Physics, 103 (1992), pp. 141-159.

[12] Jonas, S.; Frühauf, H.H.; Knab, O.: *Fully Coupled Approach to the Calculation of Nonequilibrium Hypersonic Flows Using a Godunov-Type Method.* 1st European Computational Fluid Dynamics Conf., Brussels, Belgium, September 7-11, 1992.

[13] Merci, B.; Steelant, J.; Vierendeels, J.; Riemslagh, K.; Dick, E.: *Computational Treatment of Source Terms in Two-Equation Turbulence Models.* AIAA Journal, 38 (2000), pp. 2085-2093.

[14] Courant, R.; Friedrichs, K.O.; Lewy, H.: *Über die partiellen Differenzengleichungen der mathematischen Physik.* Math. Ann., 100 (1928), pp. 32-74. Transl.: *On the Partial Difference Equations of Mathematical Physics.* IBM Journal, 11 (1967), pp. 215-234.

[15] Rizzi, A.; Inouye, M.: *A Time-Split Finite Volume Technique for Three-Dimensional Blunt-Body Flow.* AIAA Paper 73-0133, 1973.

[16] Müller, B.; Rizzi, A.: *Runge-Kutta Finite-Volume Simulation of Laminar Transonic Flow over a NACA0012 Airfoil Using the Navier-Stokes Equations.* FFA TN 1986-60, 1986.

[17] Swanson, R.C.; Turkel, E.; White, J.A.: *An Effective Multigrid Method for High-Speed Flows.* Proc. 5th Copper Mountain Conf. on Multigrid Methods, 1991.

[18] Vijayan, P.; Kallinderis, Y.: *A 3D Finite-Volume Scheme for the Euler Equations on Adaptive Tetrahedral Grids.* J. Computational Physics, 113 (1994), pp. 249-267.

[19] George, A.; Liu, J.W.: *Computer Solution of Large Sparse Positive Definite Systems.* Prentice Hall Series in Comput. Math., Englewood Cliffs, N.J., 1981.

[20] Pothen, A.; Simon, H.D.; Liou, K.P.: *Partitioning Sparse Matrices with Eigenvectors of Graphs.* SIAM J. Matrix Anal. Appl., 11 (1990), pp. 430-452.

[21] Vanden, K.J.; Whitfield, D.L.: *Direct and Iterative Algorithms for the Three-Dimensional Euler Equations.* AIAA Paper 93-3378, 1993; also AIAA Journal, 33 (1995), pp. 851-858.

[22] Cuthill, E.; McKee, J.: *Reducing the Bandwidth of Sparse Symmetric Matrices.* Proc. ACM 24th National Conference, 1969, pp. 157-161.

[23] Gibbs, N.E.; Poole, W.G.; Stockmeyer, P.K.: *An Algorithm for Reducing the Bandwidth and Profile of a Sparse Matrix.* SIAM J. Numer. Anal., 13 (1976), pp. 236-250.

[24] Löhner, R.: *Some Useful Renumbering Strategies for Unstructured Grids.* Int. J. Numerical Methods in Engineering, 36 (1993), pp. 3259-3270.

[25] Steger, J.L.; Warming, R.F.: *Flux Vector Splitting of the Inviscid Gasdynamic Equations with Application to Finite-Difference Methods.* J. Computational Physics, 40 (1981), pp. 263-293.

[26] Liou, M.S.; Van Leer, B.: *Choice of Implicit and Explicit Operators for the Upwind Differencing Method.* AIAA Paper 88-0624, 1988.

[27] Barth, T.J.; Linton, S.W.: *An Unstructured Mesh Newton Solver for Compressible Fluid Flow and its Parallel Implementation.* AIAA Paper 95-0221, 1995.

[28] Orkwis, P.D.; Vanden, K.J.: *On the Accuracy of Numerical versus Analytical Jacobians.* AIAA Paper 94-0176, 1994; also AIAA Journal, 34 (1996), pp. 1125-1129.

[29] Barth, T.J.: *Analysis of Implicit Local Linearization Techniques for Upwind and TVD Algorithms.* AIAA Paper 87-0595, 1987.

[30] Bischof, C.; Khademi, P.; Mauer, A.; Carle, A.: *Adifor 2.0: Automatic Differentiation of Fortran 77 Programs.* IEEE Computational Science & Engineering, Fall 1996, pp. 18-32.

[31] Dennis, J.E.; Schnabel, R.B.: *Numerical Methods for Unconstrained Optimization and Nonlinear Equations.* Prentice-Hall, Englewood Cliffs, NJ, 1983.

[32] Xu, X.; Richards, B.E.: *Simplified Procedure for Numerically Approximate Jacobian Matrix Generation in Newton's Method for Solving the Navier-Stokes Equations.* GU Aero Report 9320, Dept. Aerospace Eng., University of Glasgow, 1993.

[33] Curtis, A.R.; Powell, M.D.; Reid, J.K.: *On the Estimation of Sparse Jacobian Matrices.* J. Inst. Maths. Applics., 13 (1974), pp. 117-119.

[34] Venkatakrishnan, V.: *Preconditioned Conjugate Gradient Methods for the Compressible Navier-Stokes Equations.* AIAA Paper 90-0586, 1990; also AIAA Journal, 29 (1991), pp. 1092-1100.

[35] Briley, W.R.; McDonald, H.: *Solution of the Multi-Dimensional Compressible Navier-Stokes Equations by a Generalized Implicit Method.* J. Computational Physics, 24 (1977), pp. 372-397.

[36] Beam, R.; Warming, R.F.: *An Implicit Factored Scheme for the Compressible Navier-Stokes Equations.* AIAA Journal, 16 (1978), pp. 393-402.

[37] Steger, J.L.: *Implicit Finite Difference Simulation of Flow About Arbitrary Geometries with Application to Airfoils.* AIAA Journal, 16 (1978), p. 679.

[38] Pulliam, T.H.; Steger, J.L.: *Recent Improvements in Efficiency, Accuracy and Convergence for Implicit Approximate Factorization Scheme.* AIAA Paper 85-0360, 1985.

[39] Pulliam, T.H.: *Artificial Dissipation Models for the Euler Equations.* AIAA Journal, 24 (1986), pp. 1931-1940.

[40] Pulliam, T.H.; Chaussee, D.S.: *A Diagonal Form of an Implicit Approximate Factorization Algorithm.* J. Computational Physics, 39 (1981), pp. 347-363.

[41] Lomax, H.: *Some Notes on Finite Difference Methods.* Lecture Notes, Stanford University, 1986.

[42] Pulliam, T.H.: *Implicit Methods in CFD.* Numerical Methods for Fluid Dynamics III, Oxford University Press, 1988.

[43] Rosenfeld, M.; Yassour, Y.: *The Alternating Direction Multi-Zone Implicit Method.* J. Computational Physics, 110 (1994), pp. 212-220.

[44] Jameson, A.; Turkel, E.: *Implicit Scheme and LU-Decompositions.* Mathematics of Computation 37 (1981), pp. 385-397.

[45] Yoon, S.; Jameson, A.: *A Multigrid LU-SSOR Scheme for Approximate Newton-Iteration Applied to the Euler Equations.* NASA CR-17954, 1986.

[46] Yoon, S.; Jameson, A.: *Lower-Upper Symmetric-Gauss-Seidel Method for the Euler and Navier-Stokes Equations.* AIAA Paper 87-0600,1987; AIAA Journal 26 (1988), pp. 1025-1026.

[47] Yoon, S.; Jameson, A.: *LU Implicit Schemes with Multiple Grids for the Euler Equations.* AIAA Paper 86-0105, 1986; also AIAA Journal 7 (1987), pp. 929-935.

[48] Rieger, H.; Jameson, A.: *Solution of Steady 3-D Compressible Euler and Navier-Stokes Equations by an Implicit LU Scheme.* AIAA Paper 88-0619, 1988.

[49] Yoon, S.; Kwak, D.: *3-D Incompressible Navier-Stokes Solver using Lower-Upper Symmetric-Gauss-Seidel Algorithm.* AIAA Journal 29 (1991).

[50] Blazek, J.: *Investigations of the Implicit LU-SSOR Scheme.* DLR Research Report, No. 93-51, 1993.

[51] Pahlke, K.; Blazek, J.; Kirchner, A.: *Time-Accurate Euler Computations for Rotor Flows.* European Forum: Recent Developments and Applications in Aeronautical CFD, Paper No. 15, Bristol, UK, 1993.

[52] Yoon, S.; Kwak, D.: *Multigrid Convergence of an Implicit Symmetric Relaxation Scheme.* AIAA Paper 93-3357, 1993.

[53] Blazek, J.: *A Multigrid LU-SSOR Scheme for the Solution of Hypersonic Flow Problems.* AIAA Paper 94-0062, 1994.

[54] Candler, G.V.; Wright, M.J., McDonald, J.D.: *A Data-Parallel LU-SGS Method for Reacting Flows.* AIAA Paper 94-0410, 1994.

[55] Stoll, P.; Gerlinger, P.; Brüggemann, D.: *Domain Decomposition for an Implicit LU-SGS Scheme using Overlapping Grids.* AIAA Paper 97-1896, 1997.

[56] Tomaro, R.F.; Strang, W.Z.; Sankar, L.N.: *An Implicit Algorithm for Solving Time Dependent Flows on Unstructured Grids.* AIAA Paper 97-0333, 1997.

[57] Sharov, D.; Nakahashi, K.: *Reordering of 3-D Hybrid Unstructured Grids for Vectorized LU-SGS Navier-Stokes Calculations.* AIAA Paper 97-2102, 1997.

[58] Kano, S.; Kazuhiro, N.: *Navier-Stokes Computations of HSCT Off-Design Aerodynamics Using Unstructured Hybrid Grids.* AIAA Paper 98-0232, 1998.

[59] Sharov, D.; Nakahashi, K.: *Low Speed Preconditioning and LU-SGS Scheme for 3-D Viscous Flow Computations on Unstructured Grids.* AIAA Paper 98-0614, 1998.

[60] Strang, W.Z.; Tomaro, R.F.; Grismer, M.J.: *The Defining Methods of* $Cobalt_{60}$*: a Parallel, Implicit, Unstructured Euler/Navier-Stokes Flow Solver.* AIAA Paper 99-0786, 1999.

[61] Shuen, S.; Yoon, S.: *Numerical Study of Chemically Reacting Flows Using a Lower-Upper Symmetric Successive Overrelaxation Scheme.* AIAA Journal, 27 (1989), pp. 1752-1760.

[62] Shuen, J.S.: *Upwind Differencing and LU Factorization for Chemical Non-Equilibrium Navier-Stokes Equations.* J. Computational Physics, 99 (1992), pp. 233-250.

[63] Shuen, J.S.; Chen, K.H.; Choi, Y.: *A Coupled Implicit Method for Chemical Non-Equilibrium Flows at All Speeds.* J. Computational Physics, 106 (1993), pp. 306-318.

[64] Palmer, G.; Venkapathy, E.: *Comparison of Nonequilibrium Solution Algorithms Applied to Chemically Stiff Hypersonic Flows.* AIAA Journal, 33 (1995), pp. 1211-1219.

[65] Gerlinger, P.; Stoll, P.; Brüggemann, D.: *An Implicit Multigrid Method for the Simulation of Chemically Reacting Flows.* J. Computational Physics, 146 (1998), pp. 322-345.

[66] Janus, J.M.: *A Matrix Study of Ax=b for Implicit Factored Methods.* AIAA Paper 98-0112, 1998.

[67] Yuan, X.; Daiguji, H.: *A New LU-Type Implicit Scheme for Three-Dimensional Compressible Navier-Stokes Equations.* 6th Int. Symposium on CFD, Lake Tahoe, 1995. Hafez, M.; Oshima, K. (eds.), Vol. III, Pergamon Press, 1998, pp. 1473-1478.

[68] Soetrisno, M.; Imlay, S.T.; Roberts, D.W.: *A Zonal Implicit Procedure for Hybrid Structured-Unstructured Grids.* AIAA Paper 94-0645, 1994.

[69] Soetrisno, M.; Imlay, S.T.; Roberts, D.W.; Taflin, D.E.: *Development of a Zonal Implicit Procedure for Hybrid Structured-Unstructured Grids.* AIAA Paper 96-0167, 1996.

[70] Sonneveld, P.: *CGS, A Fast Lanczos-Type Solver for Nonsymmetric Linear Systems.* SIAM J. Scientific and Statistical Computing, 10 (1989), pp. 36-52.

[71] Van der Vorst, H.A.: *BiCGSTAB: A Fast and Smoothly Converging Variant of Bi-CG for the Solution of Nonsymmetric Linear Systems.* SIAM J. Scientific and Statistical Computing, 13 (1992), pp. 631-644.

[72] Freund, R.W.: *A Transpose-Free Quasi-Minimal Residual Algorithm for Non-Hermitian Linear Systems.* SIAM J. Scientific Computing, 14 (1993), pp. 470-482.

[73] Saad, Y.; Schulz, M.H.: *GMRES: A Generalized Minimum Residual Algorithm for Solving Nonsymmetric Linear Systems.* SIAM J. Sci. Stat. Comp. 7 (1986), pp. 856-869.

[74] Saad, Y.: *Krylov Subspace Techniques, Conjugate Gradients, Preconditioning and Sparse Matrix Solvers.* VKI Lecture Series, 1994-05, 1994.

[75] Walker, H.F.: *Implementation of the GMRES Method Using Householder Transformations.* SIAM J. Scientific Computing, 9 (1988), p. 152.

[76] Morgan, R.B.: *A Restarted GMRES Method Augmented with Eigenvectors.* SIAM J. Matrix Anal. Appl., 16 (1995), pp. 1154-1171.

[77] Chapman, A.; Saad, Y.: *Deflated and Augmented Krylov Subspace Techniques.* Technical Report UMSI 95/181, Minnesota Supercomputer Institute, 1995.

[78] Saad, Y.: *Enhanced Acceleration and Reconditioning Techniques.* CFD Review 1998. Hafez, M.; Oshima, K. (eds.), World Scientific Publishing Co., 1998, pp. 478-487.

[79] Ajmani, K.; Ng, W.-F.; Liou, M.-S.: *Preconditioned Conjugate Gradient Methods for Low Speed Flow Calculations.* AIAA Paper 93-0881, 1993.

[80] Brown, P.N.; Saad, Y.: *Hybrid Krylov Methods for Nonlinear Systems of Equations.* SIAM J. Scientific and Statistical Computing, 11 (1990), pp. 450-481.

[81] Zingg, D.; Pueyo, A.: *An Efficient Newton-GMRES Solver for Aerodynamic Computations.* AIAA Paper 97-1955, 1997.

[82] Gropp, W.D.: Keyes, D.E.; McInnes, L.C.; Tidriri, M.D.: *Globalized Newton-Krylov-Schwarz Algorithms and Software for Parallel Implicit CFD.* ICASE Report No. 98-24, 1998.

[83] Meijerink, J.A.; Van der Vorst, H.A.: *Guidelines for Usage of Incomplete Decompositions in Solving Sets of Linear Equations as they occur in Practical Problems.* J. Computational Physics, 44 (1981), pp. 134-155.

[84] Hackbusch, W.: *Iterative Solution of Large Sparse Systems of Equations.* Springer Verlag, New York, 1994.

[85] Venkatakrishnan, V.; Mavriplis, D.J.: *Implicit Solvers for Unstructured Meshes.* J. Computational Physics, 105 (1993), pp. 83-91.

[86] Ajmani, K.; Liou, M.-S.: *Implicit Conjugate Gradient Solvers on Distributed-Memory Architectures.* AIAA Paper 95-1695, 1995.

[87] Luo, H.; Baum, J.D.; Löhner, R.: *A Fast, Matrix-Free Implicit Method for Compressible Flows on Unstructured Grids.* AIAA Paper 99-0936, 1999.

[88] Mulder, W.; Van Leer, B.: *Experiments with Implicit Upwind Methods for the Euler Equations.* J. Computational Physics, 59 (1985), pp. 232-246.

[89] Tam, C.K.W.: *Computational Aeroacoustics: Issues and Methods.* AIAA Paper 95-0677, 1995.

[90] Tam, C.K.W.: *Applied Aero-Acoustics: Prediction Methods.* VKI Lecture Series 1996-04, 1996.

[91] Jameson, A.: *Time-Dependent Calculations Using Multigrid with Applications to Unsteady Flows Past Airfoils and Wings.* AIAA Paper 91-1596, 1991.

[92] Arnone, A.; Liou, M.S.; Povinelli, L.A.: *Multigrid Time-Accurate Integration of Navier-Stokes Equations.* AIAA Paper 93-3361, 1993.

[93] Melson, N.D.; Sanetrik, M.D.; Atkins, H.L.: *Time-Accurate Navier-Stokes Calculations with Multigrid Acceleration.* Proc. 6th Copper Mountain Conf. on Multigrid Methods, 1993, pp. 423-439.

[94] Venkatakrishnan, V.: *Implicit Schemes and Parallel Computing in Unstructured Grid CFD.* ICASE Report, No. 95-28, 1995.

[95] Venkatakrishnan, V.; Mavriplis, D.J.: *Implicit Method for the Computation of Unsteady Flows on Unstructured Grids.* J. Computational Physics, 127 (1996), pp. 380-397.

[96] Alonso, J.J.; Jameson, A.: *Fully-Implicit Time-Marching Aeroelastic Solution.* AIAA Paper 94-0056, 1994.

[97] Belov, A.; Martinelli, L.; Jameson, A.: *A New Implicit Algorithm with Multigrid for Unsteady Incompressible Flow Calculations.* AIAA Paper 95-0049, 1995.

[98] Pierce, N.A.; Alonso, J.J.: *A Preconditioned Implicit Multigrid Algorithm for Parallel Computation of Unsteady Aeroelastic Compressible Flows.* AIAA Paper 97-0444, 1997.

[99] Singh, K.P.; Newman, J.C.; Baysal, O.: *Dynamic Unstructured Method for Flows Past Multiple Objects in Relative Motion.* AIAA Journal, 33 (1995), pp. 641-649.

[100] Dubuc, L.; Cantariti, F.; Woodgate, M.; Gribben, B.; Badcock, K.J.; Richards, B.E.: *Solution of the Unsteady Euler Equations Using an Implicit Dual-Time Method.* AIAA Journal, 36 (1998), pp. 1417-1424.

[101] Pulliam, T.H.: *Time Accuracy and the Use of Implicit Methods.* AIAA paper 93-3360, 1993.

[102] Bartels, R.E.: *An Elasticity-Based Mesh Scheme Applied to the Computation of Unsteady Three-Dimensional Spoiler and Aeroelastic Problems.* AIAA Paper 99-3301, 1999.

Chapter 7

Turbulence Modelling

The outstanding feature of a turbulent flow, in the opposite to a laminar flow, is that the molecules move in a chaotic fashion along complex irregular paths. The strong chaotic motion causes the various layers of the fluid to mix together intensely. Because of the increased momentum and energy exchange between the molecules and solid walls, turbulent flows lead at the same conditions to higher skin friction and heat transfer as compared to laminar flows.

Although the chaotic fluctuations of the flow variables are of deterministic nature, the simulation of turbulent flows still continues to present a significant problem. Despite the performance of modern supercomputers, a direct simulation of turbulence by the time-dependent Navier-Stokes equations (2.19) – known as the *Direct Numerical Simulation* (DNS) [1]-[10] – is applicable only to relatively simple flow problems at low Reynolds numbers (Re). A more widespread utilisation of the DNS is prevented by the fact that the number of grid points needed for sufficient spatial resolution scales as $Re^{9/4}$ and the CPU-time as Re^3. Therefore, we are forced to account for the effects of turbulence in an approximate manner. For this purpose, a large variety of *turbulence models* was developed and the research still goes on. There are five principal classes of turbulence models:

- algebraic,
- one-equation,
- multiple-equation,
- *second-order* closures (Reynolds-stress models),
- *Large-Eddy Simulation* (LES).

The first tree models belong to the so-called *first-order* closures. They are based mostly on the *eddy-viscosity* hypothesis of Boussinesq [11], [12], but for certain applications also on *non-linear eddy-viscosity* formulations. An overview of the classes of turbulence models, which are sorted according to their decreasing level of complexity, is displayed in Fig. 7.1.

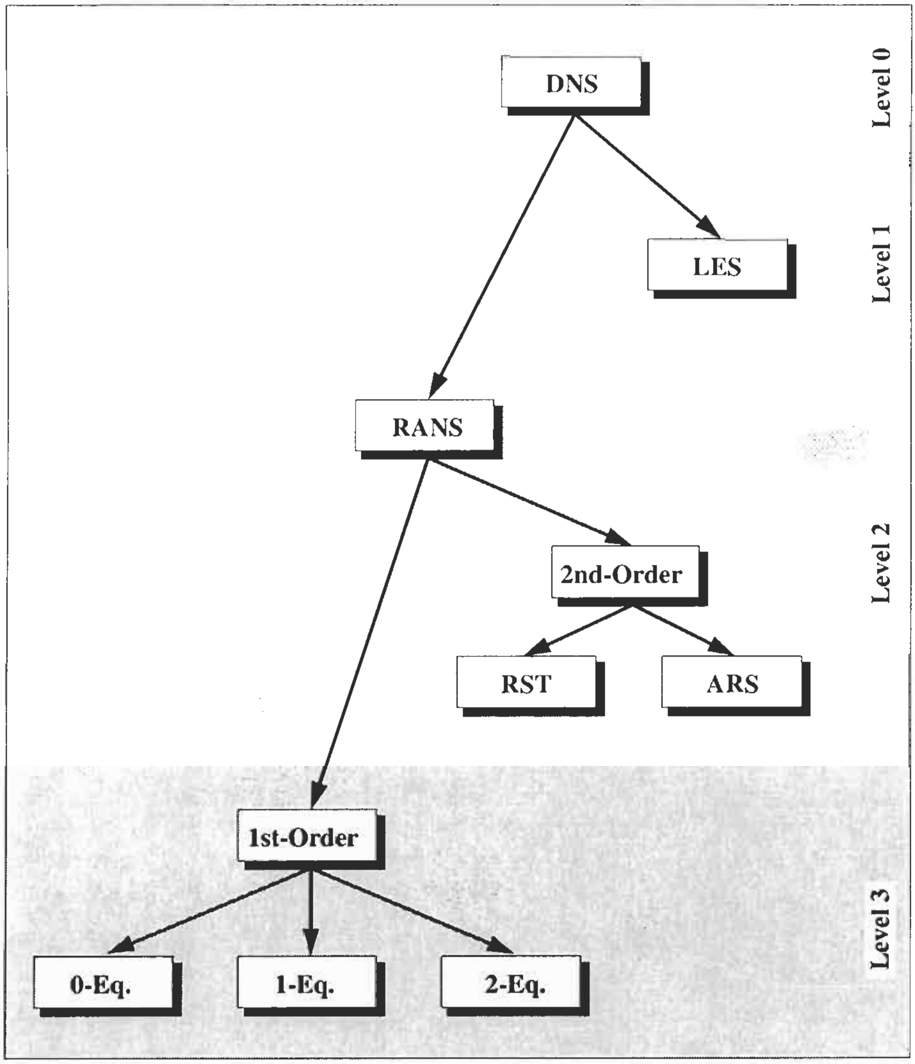


Figure 7.1: Hierarchy of turbulence models. Abbreviations:

DNS = Direct Numerical Simulation
LES = Large-Eddy Simulation
RANS = Reynolds-Averaged Navier-Stokes equations
1st-order = first-order closures
2nd-order = second-order closures
RST = Reynolds-Stress Transport models
ARS = Algebraic Reynolds-Stress models
0-, 1-, 2-Eq. = zero- (algebraic), one-, two-equations models.

One should be aware of the fact that there is no single turbulence model, which can predict reliably all kinds of turbulent flows. Each of the models has its strengths and weaknesses. For example, if a particular model works perfectly in the case of attached boundary layers, it may fail completely for separated flows. Thus, it is important always to ask whether the model includes all the significant features of the flow being investigated. Another point which should be taken into consideration is the computational effort versus the accuracy required by the particular application. We mean by this that in many cases a numerically inexpensive turbulence model can predict some global measures with the same accuracy as a more complex model.

In the following, we first introduce the basic equations of turbulence as they result from time and mass averaging of the governing equations. Then, we present the Boussinesq's and the non-linear eddy-viscosity approaches. After that, we briefly discuss the Reynolds-stress transport equation, which forms the basis of the algebraic and differential Reynolds-stress models. In Section 7.2, we present few wide-spread one- and two-equation first-order closures. Finally, we discuss the LES approach in some detail because of the growing number of engineering applications.

7.1 Basic Equations of Turbulence

First of all, let us rewrite the governing equations (2.19) in differential form (see Appendix A.1), since this is used very often in literature on turbulence modelling. Furthermore, it allows for a compact and clear notation. However, we will also provide examples of turbulence equations in integral form.

In the case of a compressible Newtonian fluid, the Navier-Stokes equations read in absence of source terms in coordinate invariant formulation as

$$\begin{aligned} \frac{\partial \rho}{\partial t} + \frac{\partial}{\partial x_i}(\rho v_i) &= 0 \\ \frac{\partial}{\partial t}(\rho v_i) + \frac{\partial}{\partial x_j}(\rho v_j v_i) &= -\frac{\partial p}{\partial x_i} + \frac{\partial \tau_{ij}}{\partial x_j} \\ \frac{\partial}{\partial t}(\rho E) + \frac{\partial}{\partial x_j}(\rho v_j H) &= \frac{\partial}{\partial x_j}(v_i \tau_{ij}) + \frac{\partial}{\partial x_j}\left(k\frac{\partial T}{\partial x_j}\right) . \end{aligned} \tag{7.1}$$

In above Eq. (7.1), v_i denotes a velocity component ($\vec{v} = [v_1, v_2, v_3]^T$), and x_i stands for a coordinate direction, respectively. An explanation of the compact tensor notation can be found in Appendix A.11.

The components of the *viscous stress* tensor τ_{ij} in Eq. (7.1) are defined as

$$\tau_{ij} = 2\mu S_{ij} + \lambda \frac{\partial v_k}{\partial x_k}\delta_{ij} = 2\mu S_{ij} - \left(\frac{2\mu}{3}\right)\frac{\partial v_k}{\partial x_k}\delta_{ij} , \tag{7.2}$$

where we utilised the Stokes's hypothesis (Eq. (2.17)). In Cartesian coordinates, Eq. (7.2) is equivalent to Eq. (2.15). The second term in Eq. (7.2), i.e., $\partial v_k/\partial x_k$, which corresponds to the divergence of the velocity, disappears for incompressible flows. The components of the *strain-rate tensor* are given by

$$S_{ij} = \frac{1}{2}\left(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i}\right) . \tag{7.3}$$

In this connection, let us also define the *rotation-rate tensor* (antisymmetric part of the velocity gradient tensor) with the following components

$$\Omega_{ij} = \frac{1}{2}\left(\frac{\partial v_i}{\partial x_j} - \frac{\partial v_j}{\partial x_i}\right) . \tag{7.4}$$

The total energy E and the total enthalpy H in Eq. (7.1) are obtained from the formulae

$$E = e + \frac{1}{2}v_i v_i , \qquad H = h + \frac{1}{2}v_i v_i \tag{7.5}$$

which correspond in Cartesian coordinate system to Eq. (2.6) and Eq. (2.12), respectively.

For incompressible flows, we can reduce Eq. (7.1) to the form

$$
\begin{aligned}
&\frac{\partial v_i}{\partial x_i} = 0 \\
&\frac{\partial v_i}{\partial t} + v_j \frac{\partial v_i}{\partial x_j} = -\frac{1}{\rho}\frac{\partial p}{\partial x_i} + \nu \nabla^2 v_i \\
&\frac{\partial T}{\partial t} + v_j \frac{\partial T}{\partial x_j} = k \nabla^2 T
\end{aligned}
\tag{7.6}
$$

with $\nu = \mu/\rho$ being the kinematic viscosity coefficient and ∇^2 denoting the Laplace operator. In the absence of buoyancy effects, the equation for the temperature T becomes decoupled from the mass conservation and momentum equations.

7.1.1 Reynolds Averaging

The first approach for the approximate treatment of turbulent flows was presented by Reynolds in 1895. The methodology is based on the decomposition of the flow variables into a mean and a fluctuating part. The governing equations (7.1) are then solved for the mean values, which are the most interesting for engineering applications. Thus, considering first incompressible flows, the velocity components and the pressure in Eq. (7.1) are substituted by [13]

$$
v_i = \overline{v}_i + v_i' \,, \qquad p = \overline{p} + p' \,, \tag{7.7}
$$

where the mean value is denoted by an overbar and the turbulent fluctuations by a prime. The mean values are obtained by an averaging procedure. There are three different forms of the *Reynolds averaging*:

1. *Time averaging* – appropriate for stationary turbulence (statistically steady turbulence)

   $$
   \overline{v}_i = \lim_{T\to\infty} \frac{1}{T} \int_t^{t+T} v_i \, dt \,. \tag{7.8}
   $$

   As a consequence, the mean value $\overline{v}_i$ does not vary in time, but only in space. The situation is sketched in Fig. 7.2. In practice, $T \to \infty$ means that the time interval T should be large as compared to the typical time-scale of the turbulent fluctuations.

2. *Spatial averaging* – appropriate for homogeneous turbulence

   $$
   \overline{v}_i = \lim_{\Omega\to\infty} \frac{1}{\Omega} \int_\Omega v_i \, d\Omega \tag{7.9}
   $$

   with Ω being a control volume. In this case, $\overline{v}_i$ is uniform in space, but it is allowed to vary in time.

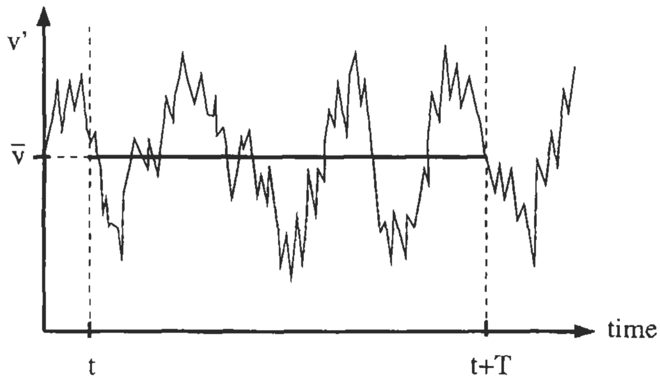


Figure 7.2: Reynolds averaging – illustration of turbulent velocity fluctuations v' and statistical mean value $\overline{v}$.

3. *Ensemble averaging* – appropriate for general turbulence

$$\overline{v}_i = \lim_{N\to\infty} \frac{1}{N} \sum_{m=1}^{N} v_i\,. \tag{7.10}$$

Here, the mean value $\overline{v}_i$ still remains a function of time and of space coordinates.

For all three approaches, the average of the fluctuating part is zero, i.e., $\overline{v_i'} = 0$. However, it can be easily seen that $\overline{v_i'v_i'} \neq 0$. The same is true for $\overline{v_i'v_j'}$, if both turbulent velocity components are correlated.

In cases where the turbulent flow is both stationary and homogeneous, all three averaging forms are equivalent. This is called the *ergodic hypothesis.*

7.1.2 Favre (Mass) Averaging

In cases where the density is not constant, it is advisable to apply the *density (mass) weighted* or Favre decomposition [14], [15] to certain quantities in Eq. (7.1) instead of Reynolds averaging. Otherwise, the averaged governing equations would become considerably more complicated due to additional correlations involving density fluctuations. The most convenient way is to employ Reynolds averaging for density and pressure, and Favre averaging for other variables such as velocity, internal energy, enthalpy and temperature. Favre averaged quantities, for example the velocity components, are obtained from the relation [14], [15]

$$\tilde{v}_i = \frac{1}{\overline{\rho}} \lim_{T\to\infty} \frac{1}{T} \int_t^{t+T} \rho v_i \, dt\,, \tag{7.11}$$

where $\overline{\rho}$ denotes the Reynolds-averaged density. Hence, the Favre decomposition reads

$$v_i = \tilde{v}_i + v_i''\,, \tag{7.12}$$

where $\tilde{v}_i$ represents the mean value and v_i'' the fluctuating part of the velocity v_i. Again, the average of the fluctuating part is zero, i.e., $\widetilde{v_i''} = 0$. Furthermore, the average of the product of two fluctuating quantities is not zero, if the quantities are correlated. Hence, for example, $\widetilde{v_i''v_i''} \neq 0$ and in general $\widetilde{v_i''v_j''} \neq 0$.

The following relationships can be derived for a mix between Favre and Reynolds averaging

$$\widetilde{\rho v_i} = \overline{\rho}\,\tilde{v}_i\,, \quad \overline{\rho v_i''} = 0\,, \quad \text{but } \overline{v_i''} \neq 0\,. \tag{7.13}$$

These equations will be utilised in later subsections.

7.1.3 Reynolds-Averaged Navier-Stokes Equations

If we apply either the time averaging Eq. (7.8) or the ensemble averaging Eq. (7.10) to the incompressible Navier-Stokes equations (7.6), we obtain the following relations for the mass and momentum conservation

$$\begin{aligned} \frac{\partial \overline{v}_i}{\partial x_i} &= 0 \\ \rho\frac{\partial \overline{v}_i}{\partial t} + \rho\overline{v}_j\frac{\partial \overline{v}_i}{\partial x_j} &= -\frac{\partial \overline{p}}{\partial x_i} + \frac{\partial}{\partial x_j}\left(\overline{\tau}_{ij} - \rho\,\overline{v_i'v_j'}\right)\,. \end{aligned} \tag{7.14}$$

These are known as the *Reynolds-Averaged Navier-Stokes equations* (RANS). The equations (7.14) are formally identical to the Navier-Stokes equations (7.1) or (7.6) with the exception of the additional term

$$\tau_{ij}^R = -\rho\,\overline{v_i'v_j'} = -\rho\left(\overline{v_i v_j} - \overline{v}_i\overline{v}_j\right)\,, \tag{7.15}$$

which constitutes the so-called *Reynolds-stress tensor.* It represents the transfer of momentum due to turbulent fluctuations. The laminar viscous stresses are evaluated according to Eqs. (7.2) and (7.3) using Reynolds-averaged velocity components, i.e.,

$$\overline{\tau}_{ij} = 2\mu\,\overline{S}_{ij} = \mu\left(\frac{\partial \overline{v}_i}{\partial x_j} + \frac{\partial \overline{v}_j}{\partial x_i}\right)\,. \tag{7.16}$$

The Reynolds-stress tensor consists in 3D of nine components

$$\rho\,\overline{v_i'v_j'} = \begin{bmatrix} \rho\overline{(v_1')^2} & \rho\overline{v_1'v_2'} & \rho\overline{v_1'v_3'} \\ \rho\overline{v_2'v_1'} & \rho\overline{(v_2')^2} & \rho\overline{v_2'v_3'} \\ \rho\overline{v_3'v_1'} & \rho\overline{v_3'v_2'} & \rho\overline{(v_3')^2} \end{bmatrix}\,. \tag{7.17}$$

However, since v_i' and v_j' in the correlations can be interchanged, the Reynolds-stress tensor contains only six independent components. The sum of the normal stresses divided by density defines the *turbulent kinetic energy*, i.e.,

$$K = \frac{1}{2}\,\overline{v_i'v_i'} = \frac{1}{2}\left[\overline{(v_1')^2} + \overline{(v_2')^2} + \overline{(v_3')^2}\right]\,. \tag{7.18}$$

As we can see, the fundamental problem of turbulence modelling based on the Reynolds-averaged Navier-Stokes equations is to find six additional relations in order to close the equations (7.14). We shall introduce the basic methodologies in the Subsections 7.1.5-7.1.7.

7.1.4 Favre- and Reynolds-Averaged Navier-Stokes Equations

In turbulence modelling, it is quite common to assume that *Morkovin's hypothesis* [16] is valid. It states that the turbulent structure of a boundary layer is not notably influenced by density fluctuations if $\rho' \ll \overline{\rho}$. This is generally true for wall-bounded flows up to a Mach number of about five. However, in the case of hypersonic flows or for compressible free shear layers, density fluctuations have to be taken into account. The same holds also for flows with combustion or significant heat transfer.

Application of the Reynolds averaging (Eq. (7.8) or (7.10)) to density and pressure, and of the Favre averaging Eq. (7.11) to the remaining flow variables in the compressible Navier-Stokes equations (7.1) yields [17]

$$\begin{aligned}
&\frac{\partial \overline{\rho}}{\partial t} + \frac{\partial}{\partial x_i}(\overline{\rho}\,\tilde{v}_i) = 0 \\
&\frac{\partial}{\partial t}(\overline{\rho}\,\tilde{v}_i) + \frac{\partial}{\partial x_j}(\overline{\rho}\,\tilde{v}_j\tilde{v}_i) = -\frac{\partial \overline{p}}{\partial x_i} + \frac{\partial}{\partial x_j}\left(\tilde{\tau}_{ij} - \overline{\rho}\,\widetilde{v_i''v_j''}\right) \\
&\frac{\partial}{\partial t}(\overline{\rho}\tilde{E}) + \frac{\partial}{\partial x_j}(\overline{\rho}\,\tilde{v}_j\tilde{H}) = \frac{\partial}{\partial x_j}\left(k\frac{\partial \tilde{T}}{\partial x_j} - \overline{\rho}\,\widetilde{v_j''h''} + \widetilde{\tau_{ij}v_i''} - \overline{\rho}\,\widetilde{v_j''K}\right) \\
&\qquad + \frac{\partial}{\partial x_j}\left[\tilde{v}_i\left(\tilde{\tau}_{ij} - \overline{\rho}\,\widetilde{v_i''v_j''}\right)\right] .
\end{aligned} \tag{7.19}$$

These are the *Favre- and Reynolds-Averaged Navier-Stokes* equations. Similarly to the Reynolds averaging, the viscous stress tensor in the momentum (and energy) equation is extended by the *Favre-averaged Reynolds-stress* tensor, i.e.,

$$\tau_{ij}^F = -\overline{\rho}\,\widetilde{v_i''v_j''} . \tag{7.20}$$

Its form is similar to Eq. (7.17) with Favre instead of Reynolds averaging. The components of the laminar (molecular) viscous stress tensor $\tilde{\tau}_{ij}$ are evaluated by Eq. (7.2) using Favre-averaged velocity components.

If we employ the definition of the Favre-averaged turbulent kinetic energy, i.e.,

$$\overline{\rho}\tilde{K} = \frac{1}{2}\overline{\rho}\,\widetilde{v_i''v_i''} , \tag{7.21}$$

we can express the total energy in Eq. (7.19) as

$$\overline{\rho}\tilde{E} = \overline{\rho}\,\tilde{e} + \frac{1}{2}\overline{\rho}\,\tilde{v}_i\tilde{v}_i + \frac{1}{2}\overline{\rho}\,\widetilde{v_i''v_i''} = \overline{\rho}\,\tilde{e} + \frac{1}{2}\overline{\rho}\,\tilde{v}_i\tilde{v}_i + \overline{\rho}\tilde{K} . \tag{7.22}$$

The total enthalpy is defined as

$$\overline{\rho}\tilde{H} = \overline{\rho}\,\tilde{h} + \frac{1}{2}\overline{\rho}\,\tilde{v}_i\tilde{v}_i + \frac{1}{2}\overline{\rho}\,\widetilde{v_i''v_i''} = \overline{\rho}\,\tilde{h} + \frac{1}{2}\overline{\rho}\,\tilde{v}_i\tilde{v}_i + \overline{\rho}\tilde{K}\,. \tag{7.23}$$

The individual parts of the Favre- and Reynolds-averaged Navier-Stokes equations (7.19) have the following physical meaning [17]:

$\frac{\partial}{\partial x_j}\left(k\frac{\partial \tilde{T}}{\partial x_j}\right)$ – molecular diffusion of heat

$\frac{\partial}{\partial x_j}(\overline{\rho}\,\widetilde{v_j''h''})$ – turbulent transport of heat

$\frac{\partial}{\partial x_j}(\widetilde{\tau_{ij}v_i''})$ – molecular diffusion of $\tilde{K}$

$\frac{\partial}{\partial x_j}(\overline{\rho}\,\widetilde{v_j''K})$ – turbulent transport of $\tilde{K}$

$\frac{\partial}{\partial x_j}(\tilde{v}_i\tilde{\tau}_{ij})$ – work done by the molecular stresses

$\frac{\partial}{\partial x_j}(\tilde{v}_i\tau_{ij}^F)$ – work done by the Favre-averaged Reynolds stresses

The molecular diffusion and turbulent transport of $\tilde{K}$ are very often neglected. This is a valid approximation for transonic and supersonic flows. In order to close the Favre- and Reynolds-averaged equations (7.19), we also have to supply six components of the Favre-averaged Reynolds-stress tensor (Eq. (7.20)) and three components of the turbulent heat-flux vector. We shall discuss the three basic approaches in the next subsections.

7.1.5 Eddy-Viscosity Hypothesis

One of the most significant contributions to turbulence modelling was presented in 1877 by Boussinesq [11], [12]. His idea is based on the observation that the momentum transfer in a turbulent flow is dominated by the mixing caused by large energetic turbulent eddies. The Boussinesq hypothesis assumes that the turbulent shear stress is related linearly to mean rate of strain, as in a laminar flow. The proportionality factor is the *eddy viscosity*. The Boussinesq hypothesis for Reynolds averaged incompressible flow (Eq. (7.14)) can be written as

$$\tau_{ij}^R = -\rho\,\overline{v_i'v_j'} = 2\mu_T\overline{S}_{ij} - \frac{2}{3}\rho K\delta_{ij}\,, \tag{7.24}$$

where $\overline{S}_{ij}$ denotes the Reynolds-averaged strain-rate tensor (Eq. (7.3), cf. also Eq. (7.16)), K is the turbulent kinetic energy ($K = (1/2)\overline{v_i'v_i'}$), and μ_T stands for the eddy viscosity. Unlike the molecular viscosity μ, the eddy viscosity μ_T represents no physical characteristic of the fluid, but it is a function of the local flow conditions. Additionally, μ_T is also strongly affected by flow history effects.

In the case of the compressible Favre- and Reynolds-averaged Navier-Stokes equations (7.19), the Boussinesq eddy-viscosity hypothesis reads

$$\tau_{ij}^F = -\overline{\rho}\,\widetilde{v_i''v_j''} = 2\mu_T\tilde{S}_{ij} - \left(\frac{2\mu_T}{3}\right)\frac{\partial\tilde{v}_k}{\partial x_k}\delta_{ij} - \frac{2}{3}\overline{\rho}\tilde{K}\delta_{ij}\,, \tag{7.25}$$

where $\tilde{S}_{ij}$ and $\tilde{K}$ are the Favre-averaged strain rate and turbulent kinetic energy, respectively. Note the similarity to Eq. (7.2). The term $(2/3)\rho K\delta_{ij}$ in Eqs. (7.24) and (7.25) is required in order to obtain the proper trace of τ_{ij}^R or τ_{ij}^F. This means that we must have

$$\tau_{ii}^R = -2\rho K \quad \text{or} \quad \tau_{ii}^F = -2\,\overline{\rho}\tilde{K}$$

in the case of $\overline{S}_{ii} = 0$ (continuity equation) or $\tilde{S}_{ii} = 0$, in order to fulfil the relations Eq. (7.18) or (7.21) for the turbulent kinetic energy. However, the term $(2/3)\rho K\delta_{ij}$ is often neglected, particularly in connection with simpler turbulence models (like algebraic ones).

The approximation, which is commonly used for the modelling of the turbulent heat-flux vector, is based on the classical Reynolds analogy [18]. Hence, we may write

$$\overline{\rho}\,\widetilde{v_j''h''} = -k_T\frac{\partial\tilde{T}}{\partial x_j} \tag{7.26}$$

with the *turbulent thermal conductivity coefficient* k_T being defined as

$$k_T = c_p\frac{\mu_T}{Pr_T}\,. \tag{7.27}$$

In Equation (7.27), c_p denotes the specific heat coefficient at constant pressure and Pr_T is the turbulent Prandtl number. The turbulent Prandtl number is in general assumed to be constant over the flow field ($Pr_T = 0.9$ for air).

By applying the eddy-viscosity approach to the Reynolds- (and Favre-) averaged form of the governing equations (2.19) or Eq. (7.1), the dynamic viscosity coefficient μ in the viscous stress tensor Eq. (2.15) or Eq. (7.2) is simply replaced by the sum of a laminar and a turbulent component, i.e.,

$$\mu = \mu_L + \mu_T\,. \tag{7.28}$$

The laminar viscosity μ_L is computed, for example, with the aid of the Sutherland formula (2.30). Furthermore, according to the Reynolds analogy given by Eq. (7.26), the thermal conductivity coefficient k in Eq. (2.24) or Eq. (7.2) is evaluated as

$$k = k_L + k_T = c_p\left(\frac{\mu_L}{Pr_L} + \frac{\mu_T}{Pr_T}\right)\,. \tag{7.29}$$

The eddy-viscosity concept of Boussinesq is, at least from engineering point of view, very attractive since it requires "only" the determination of μ_T (the turbulent kinetic energy K needed for the term $(2/3)\rho K\delta_{ij}$ in Eq. (7.24) or (7.25) is either obtained as a by-product of the turbulence model or is simply omitted).

Once we know the eddy viscosity μ_T, we can easily extend the Navier-Stokes equations (2.19) or (7.1) to simulate turbulent flow by introducing averaged flow variables and by adding μ_T to the laminar viscosity. Therefore, Boussinesq's approach became the basis for a large variety of first-order turbulence closures. However, there are applications for which the Boussinesq hypothesis is no longer valid (see, e.g., [17] p. 214 or [19] p. 111):

- flows with sudden change of mean strain rate,
- flows with significant streamline curvature,
- flows with rotation and stratification,
- secondary flows in ducts and in turbomachinery,
- flows with boundary layer separation and reattachment.

The limitations of the eddy-viscosity approach are caused by the assumption of equilibrium between the turbulence and the mean strain field, as well as by the independence on system rotation. The results can be notably improved by using appropriate correction terms in the turbulence models [20], [21]. Further increased accuracy of predictions can be achieved through the application of non-linear eddy-viscosity models which are described next.

7.1.6 Non-Linear Eddy Viscosity

In order to remove the restrictions imposed by the assumption of equilibrium between the turbulence and the mean strain rate, Lumley [22], [23] proposed to extend the linear Boussinesq approach by higher-order products of strain and rotation tensors. This can viewed as a Taylor series expansion. Following the idea of Lumley, numerous non-linear eddy-viscosity models were proposed, see, for example, Refs. [24]–[27].

In the following, we shall present one recent approach proposed by Shih et al. [25]. It includes up to third-order terms in the general eddy-viscosity formulation and is particularly suited to swirling flows. As already pointed out in [19], p. 194, cubic terms are essential for high accuracy. The Reynolds stresses τ_{ij}^R can be expressed as [25], [28] (cf. Eq. (7.24))

$$
\begin{aligned}
\rho\,\overline{v_i'v_j'} = {} & \frac{2}{3}\rho K\delta_{ij} - C_1\frac{\rho K^2}{\varepsilon}\,2\,S_{ij}^* - C_3\frac{\rho K^3}{\varepsilon^2}\left[\overline{S}_{ik}\overline{\Omega}_{kj} - \overline{\Omega}_{ik}\overline{S}_{kj}\right] \\
& - C_4\frac{\rho K^4}{\varepsilon^3}\left[(\overline{S}_{ik})^2\,\overline{\Omega}_{kj} - \overline{\Omega}_{ik}(\overline{S}_{kj})^2\right] \\
& + C_5\frac{\rho K^4}{\varepsilon^3}\left[\overline{\Omega}_{ik}\overline{S}_{km}\overline{\Omega}_{mj} - \frac{1}{3}\overline{\Omega}_{kl}\overline{S}_{lm}\overline{\Omega}_{mk}\delta_{ij} + I_s S_{ij}^*\right]
\end{aligned}
\qquad (7.30)
$$

with $\overline{S}_{ij}$, $\overline{\Omega}_{ij}$ according to Eqs. (7.3) and (7.4). Furthermore,

$$I_s = \frac{1}{2}\left[\overline{S}_{kk}\overline{S}_{ll} - (\overline{S}_{kk})^2\right] S^*_{ij}$$
$$S^*_{ij} = \overline{S}_{ij} - \frac{1}{3}\overline{S}_{kk}\delta_{ij} \,. \tag{7.31}$$

The values of the turbulent kinetic energy K and the *dissipation rate* ε are obtained from low-Reynolds K-ε turbulence model (cf. Subsection 7.2.2). The factors C_1 to C_5 in Eq. (7.30) are given in [25], [28].

In comparison to the linear eddy-viscosity approach, the non-linear models are computationally only slightly more expensive, but they offer a substantially improved prediction capabilities for complex turbulent flows.

7.1.7 Reynolds-Stress Transport Equation

It is possible to derive **exact** equations for the Reynolds stresses by taking the time average (second-order moment)

$$\overline{v_i' \mathcal{N}(v_j) + v_j' \mathcal{N}(v_i)} = 0 \,, \tag{7.32}$$

where $\mathcal{N}(v_i)$ denotes the Navier-Stokes operator, i.e.,

$$\mathcal{N}(v_i) = \rho\frac{\partial v_i}{\partial t} + \rho v_j \frac{\partial v_i}{\partial x_j} + \frac{\partial p}{\partial x_i} - \mu\nabla^2 v_i \,. \tag{7.33}$$

Using the average Eq. (7.32) together with Eq. (7.33), we obtain the following *Reynolds-stress transport equation* [29]

$$\frac{\partial \tau^R_{ij}}{\partial t} + \overline{v}_k \frac{\partial \tau^R_{ij}}{\partial x_k} = P_{ij} + \Pi_{ij} - \varepsilon_{ij} - \frac{\partial C_{ijk}}{\partial x_k} + \mu\nabla^2\tau^R_{ij} \tag{7.34}$$

for incompressible flow. The formulation for compressible flows can be found in Ref. [17], p. 179, or in [30], [31]. The production of the turbulent kinetic energy P_{ij}, the pressure-strain term Π_{ij}, the dissipation-rate term ε_{ij}, and the third-order diffusion term C_{ijk} in Eq. (7.34) are defined as

$$P_{ij} = -\tau^R_{ik}\frac{\partial \overline{v}_j}{\partial x_k} - \tau^R_{jk}\frac{\partial \overline{v}_i}{\partial x_k}$$
$$\Pi_{ij} = \overline{p'\left(\frac{\partial v_i'}{\partial x_j} + \frac{\partial v_j'}{\partial x_i}\right)} = 2\,\overline{p' S'_{ij}}$$
$$\varepsilon_{ij} = 2\mu\,\overline{\frac{\partial v_i'}{\partial x_k}\frac{\partial v_j'}{\partial x_k}} \tag{7.35}$$
$$C_{ijk} = \rho\,\overline{v_i' v_j' v_k'} + \overline{p' v_i'}\,\delta_{jk} + \overline{p' v_j'}\,\delta_{ik} \,.$$

In Eq. (7.35), S'_{ij} denotes the fluctuating part of the strain-rate tensor. The first part of C_{ijk}, the triple velocity term, represents transport driven by fluctuating convection, the two other parts are the pressure transport terms (pressure-velocity correlations), respectively.

As we can see, the exact Reynolds-stress equation contains new unknown higher-order correlations (e.g., $\overline{v'_i v'_j v'_k}$). Therefore, Equation (7.34) can be closed only by using **empirical** models. This is caused by the non-linear nature of the Navier-Stokes equations. The second-order closures – the Reynolds-stress models – provide the necessary framework for solving Eq. (7.34).

7.2 First-Order Closures

The first-order closures represent the easiest way to approximate the Reynolds stresses in the Reynolds-/Favre-averaged Navier-Stokes equations. They are based on Boussinesq or non-linear eddy-viscosity models, which we discussed in the Subsections 7.1.5 and 7.1.6, respectively. Consequently, the task of an associated turbulence model is to compute the eddy viscosity μ_T.

From the large variety of first-order closure models, we selected three widely-used approaches which represent the current state-of-the-art. All three models can be implemented easily on structured as well as on unstructured grids. First, we shall discuss the one-equation model due to Spalart and Allmaras. Second, we shall present the well-known K-ε two-equation model. Finally, we shall consider the K-ω SST (Shear-Stress Transport) two-equation model proposed by Menter. A detailed comparison of this turbulence models for various cases can be found in [32].

In the following, the density and the velocity components should be understood as Reynolds-/Favre-averaged, although the corresponding notation is omitted for convenience.

7.2.1 Spalart-Allmaras One-Equation Model

The Spalart-Allmaras one-equation turbulence model [33] employs transport equation for an eddy-viscosity variable $\tilde{\nu}$. It was developed based on empiricism, dimensional analysis and Galilean invariance. It was calibrated using results for 2-D mixing layers, wakes and flat-plate boundary layers. The Spalart-Allmaras model also allows for reasonably accurate predictions of turbulent flows with adverse pressure gradients. Furthermore, it is capable of smooth transition from laminar to turbulent flow at user specified locations. The Spalart-Allmaras model has several favourable numerical features. It is "local" which means that the equation at one point does not depend on the solution at other points. Therefore, it can be readily implemented on structured multi-block or on unstructured grids. It is also robust, converges fast to steady-state and requires only moderate grid resolution in the near-wall region.

Differential Form

The Spalart-Allmaras turbulence model can be written in tensor notation as follows [33]

$$
\begin{aligned}
\frac{\partial \tilde{\nu}}{\partial t} + \frac{\partial}{\partial x_j}(\tilde{\nu}\, v_j) = {} & C_{b1}(1 - f_{t2})\tilde{S}\tilde{\nu} \\
& + \frac{1}{\sigma}\left\{ \frac{\partial}{\partial x_j}\left[(\nu_L + \tilde{\nu})\frac{\partial \tilde{\nu}}{\partial x_j}\right] + C_{b2}\frac{\partial \tilde{\nu}}{\partial x_j}\frac{\partial \tilde{\nu}}{\partial x_j} \right\} \\
& - \left[C_{w1} f_w - \frac{C_{b1}}{\kappa^2} f_{t2}\right]\left(\frac{\tilde{\nu}}{d}\right)^2 + f_{t1}\|\Delta \vec{v}\|_2^2 \,.
\end{aligned}
\tag{7.36}
$$

The terms on the right-hand side represent eddy-viscosity production, conservative diffusion, non-conservative diffusion, near-wall turbulence destruction, transition damping of production, and transition source of turbulence. Furthermore, $\nu_L = \mu_L/\rho$ denotes the laminar kinematic viscosity and d is the distance to the closest wall. The turbulent eddy viscosity in Eq. (7.28) and (7.29) is obtained from the formula

$$\mu_T = f_{v1}\,\rho\,\tilde{\nu}\,. \tag{7.37}$$

The production term is evaluated with the following formulae

$$\begin{aligned}
\tilde{S} &= f_{v3}\,S + \frac{\tilde{\nu}}{\kappa^2 d^2}\,f_{v2}\,, \\
f_{v1} &= \frac{\chi^3}{\chi^3 + C_{v1}^3}\,, \quad f_{v2} = \left(1 + \frac{\chi}{C_{v2}}\right)^{-3}, \\
f_{v3} &= \frac{(1+\chi f_{v1})(1-f_{v2})}{\max(\chi,\,0.001)}\,, \quad \chi = \frac{\tilde{\nu}}{\nu_L}
\end{aligned} \tag{7.38}$$

In Equation (7.38), S stands for the magnitude of the mean rotation rate, i.e.,

$$S = \sqrt{2\,\Omega_{ij}\Omega_{ij}}\,,$$

where Ω_{ij} is given by Eq. (7.4). Note that $\tilde{S}$ differs from its original definition in [33]. The modification was suggested by Spalart in order to prevent $\tilde{S}$ from reaching zero (cf. Ref. [34], p. 155).

The terms controlling the destruction of the eddy viscosity read

$$\begin{aligned}
f_w &= g\left(\frac{1 + C_{w3}^6}{g^6 + C_{w3}^6}\right)^{1/6}, \\
g &= r + C_{w2}(r^6 - r)\,, \quad r = \frac{\tilde{\nu}}{\tilde{S}\,\kappa^2 d^2}\,.
\end{aligned} \tag{7.39}$$

Functions used for modelling the laminar-turbulent transition are given by

$$\begin{aligned}
f_{t1} &= C_{t1} g_t \exp\left(-C_{t2}\frac{\omega_t^2}{\Delta U^2}\left(d^2 + g_t^2 d_t^2\right)\right), \\
f_{t2} &= C_{t3}\exp\left(-C_{t4}\,\chi^2\right), \quad g_t = \min\left[0.1,\, \|\Delta\vec{v}\|_2/(\omega_t\,\Delta x_t)\right],
\end{aligned} \tag{7.40}$$

where ω_t represents the vorticity at the wall at the trip point (position has to be specified by the user), $\|\Delta\vec{v}\|_2$ denotes the 2-norm of the difference between the velocity at the trip point and the current field point, d_t is the distance to the nearest trip point, and Δx_t stands for the spacing along the wall at the trip point.

Finally, the various constants in Eqs. (7.36)-(7.40) are defined as

$$
\begin{gathered}
C_{b1} = 0.1355\,, \quad C_{b2} = 0.622\,, \\
C_{v1} = 7.1\,, \quad C_{v2} = 5\,, \quad \sigma = 2/3\,, \quad \kappa = 0.41\,, \\
C_{w1} = C_{b1}/\kappa^2 + (1 + C_{b2})/\sigma\,, \quad C_{w2} = 0.3\,, \quad c_{w3} = 2\,, \\
C_{t1} = 1, \quad C_{t2} = 2, \quad C_{t3} = 1.3, \quad C_{t4} = 0.5\,.
\end{gathered}
\tag{7.41}
$$

As pointed out in [33], it is convenient to substitute the non-conservative diffusion term in Eq. (7.36), i.e.,

$$
\frac{1}{\sigma}\left\{\frac{\partial}{\partial x_j}\left[(\nu_L + \tilde{\nu})\frac{\partial \tilde{\nu}}{\partial x_j}\right] + C_{b2}\frac{\partial \tilde{\nu}}{\partial x_j}\frac{\partial \tilde{\nu}}{\partial x_j}\right\}
$$

by the following expression

$$
\frac{1 + C_{b2}}{\sigma}\frac{\partial}{\partial x_j}\left[(\nu_L + \tilde{\nu})\frac{\partial \tilde{\nu}}{\partial x_j}\right] - \frac{C_{b2}}{\sigma}(\nu_L + \tilde{\nu})\nabla^2\tilde{\nu}\,. \tag{7.42}
$$

In this way, difficulties with the discretisation of the term $(\partial\tilde{\nu}/\partial x_j)^2$ are circumvented.

Integral Form

The Spalart-Allmaras turbulence model (Eq. (7.36)) reads after the transformation into a finite volume scheme as follows

$$
\frac{\partial}{\partial t}\int_\Omega \tilde{\nu}\, d\Omega + \oint_{\partial\Omega}(F_{c,T} - F_{v,T})\, dS = \int_\Omega Q_T\, d\Omega\,, \tag{7.43}
$$

where Ω represents the control volume, $\partial\Omega$ its surface, and dS is a surface element of Ω. The convective flux is defined as

$$
F_{c,T} = \tilde{\nu}V \tag{7.44}
$$

with V being the contravariant velocity (see Eq. (2.22)). The convective flux is in general discretised using first-order upwind scheme. The viscous flux is given by

$$
F_{v,T} = n_x\tau_{xx}^T + n_y\tau_{yy}^T + n_z\tau_{zz}^T\,, \tag{7.45}
$$

where n_x, n_y, and n_z are the components of the unit normal vector. The normal viscous stresses reads

$$
\begin{gathered}
\tau_{xx}^T = \frac{1}{\sigma}(\nu_L + \tilde{\nu})\frac{\partial\tilde{\nu}}{\partial x}\,, \quad \tau_{yy}^T = \frac{1}{\sigma}(\nu_L + \tilde{\nu})\frac{\partial\tilde{\nu}}{\partial y}\,, \\
\tau_{zz}^T = \frac{1}{\sigma}(\nu_L + \tilde{\nu})\frac{\partial\tilde{\nu}}{\partial z}\,.
\end{gathered}
\tag{7.46}
$$

Finally, the source term in Eq. (7.43) becomes

$$Q_T = C_{b1}(1 - f_{t2})\tilde{S}\tilde{\nu} + \frac{C_{b2}}{\sigma}\left[\left(\frac{\partial \tilde{\nu}}{\partial x}\right)^2 + \left(\frac{\partial \tilde{\nu}}{\partial y}\right)^2 + \left(\frac{\partial \tilde{\nu}}{\partial z}\right)^2\right] - \left[C_{w1}f_w - \frac{C_{b1}}{\kappa^2}f_{t2}\right]\left(\frac{\tilde{\nu}}{d}\right)^2 + f_{t1}\|\Delta\vec{v}\|_2^2 . \tag{7.47}$$

Alternatively, the non-conservative diffusion can be formulated as suggested by Eq. (7.42). The model constants are defined in Eq. (7.41).

Initial and Boundary Conditions

The initial value of $\tilde{\nu}$ is usually taken as $\tilde{\nu} = 0.1\nu_L$. The same value is also specified at inflow boundaries. At outflow boundaries, $\tilde{\nu}$ is simply extrapolated from the interior of the computational domain. At solid walls, it is appropriate to set $\tilde{\nu} = 0$ and hence $\mu_T = 0$.

7.2.2 K-ε Two-Equation Model

The K-ε turbulence model is the most widely employed two-equation eddy-viscosity model. It is based on the solution of equations for the turbulent kinetic energy and the turbulent dissipation rate. The historic roots of the K-ε model reach to the work of Chou [35]. During the 1970's, various formulations of the model were proposed. The most important contributions were due to Jones and Launder [36], [37], Launder and Sharma [38] as well as due to Launder and Spalding [39].

The K-ε turbulence model requires addition of the so-called *damping functions* in order to stay valid through the viscous sublayer to the wall. The aim of the damping functions is to assure proper limiting behaviour of K and ε at the wall, i.e.,

$$K \sim y^2 \qquad \text{and} \qquad \frac{\varepsilon}{K} \sim \frac{2\nu}{y^2} \qquad \text{for} \quad y \to 0 \, , \tag{7.48}$$

where y represents the coordinate normal to the wall. Further, it can be shown that the Reynolds shear stress behaves like (see, e.g., [17] pp. 138-139)

$$\tau_{ij}^R \sim y^3 \qquad \text{for} \quad y \to 0 \, , \; i \neq j \, . \tag{7.49}$$

The K-ε models with damping functions are also denoted as *low Reynolds number* models. The most widely used formulations of the damping functions were proposed by Jones and Launder [36], Launder and Sharma [38], Lam and Bremhorst [40], and by Chien [41]. The reader my find a comparison of seven different low Reynolds number K-ε models in Ref. [42].

The K-ε turbulence model is more difficult to solve numerically than the previously discussed Spalart-Allmaras model (Subsection 7.2.1). Particularly,

the damping functions lead to turbulence equations with stiff source terms. This, and the necessary high grid resolution nearby walls (in order to resolve the viscous sublayer), requires the utilisation of at least point-implicit or better full-implicit time-stepping schemes. Reference [43] contains useful hints on the explicit time discretisation of the K-ε equations. Examples of implementations of the K-ε model on structured as well as on unstructured grids can be found, e.g., in [44]-[52]. Finally, it is important to note that the accuracy of the K-ε model degrades for flows with adverse pressure gradient [42], [17].

Differential Form

A low Reynolds number K-ε model can be written as

$$\frac{\partial \rho K}{\partial t} + \frac{\partial}{\partial x_j}(\rho v_j K) = \frac{\partial}{\partial x_j}\left[\left(\mu_L + \frac{\mu_T}{\sigma_K}\right)\frac{\partial K}{\partial x_j}\right] + \tau_{ij}^F S_{ij} - \rho\varepsilon$$

$$\begin{aligned}\frac{\partial \rho \varepsilon^*}{\partial t} + \frac{\partial}{\partial x_j}(\rho v_j \varepsilon^*) &= \frac{\partial}{\partial x_j}\left[\left(\mu_L + \frac{\mu_T}{\sigma_\varepsilon}\right)\frac{\partial \varepsilon^*}{\partial x_j}\right] + C_{\varepsilon 1} f_{\varepsilon 1}\frac{\varepsilon^*}{K}\tau_{ij}^F S_{ij} \\ &- C_{\varepsilon 2} f_{\varepsilon 2}\,\rho\,\frac{(\varepsilon^*)^2}{K} + \phi_\varepsilon\,.\end{aligned} \tag{7.50}$$

The terms on the right-hand side represent conservative diffusion, eddy-viscosity production and dissipation, respectively. Furthermore, ϕ_ε denotes the so-called explicit wall term. The Favre-averaged turbulent stresses τ_{ij}^F are given by Eq. (7.25) and the strain-rate tensor S_{ij} follows from Eq. (7.3). The turbulent eddy viscosity in Eq. (7.28) and (7.29) results from

$$\mu_T = C_\mu f_\mu\,\rho\,\frac{K^2}{\varepsilon^*}\,. \tag{7.51}$$

The turbulent kinetic energy is also employed for the evaluation of the eddy viscosity according to Eq. (7.24) or Eq. (7.25). The quantity ε^* is related to the turbulent dissipation rate ε by

$$\varepsilon = \varepsilon_w + \varepsilon^*\,. \tag{7.52}$$

The term ε_w is the value of the dissipation rate at the wall. The definition in Eq. (7.52) greatly simplifies the application of wall-boundary conditions (see further below).

The constants, the near-wall damping functions as well as the wall term differ between the various K-ε models. Here, we choose the Launder-Sharma model because it gives good results for a wide range of applications [42]. For the Launder-Sharma model, the constants and the turbulent Prandtl number are given by [38]

$$\begin{gathered} C_\mu = 0.09\,,\quad C_{\varepsilon 1} = 1.44\,,\quad C_{\varepsilon 2} = 1.92\,, \\ \sigma_K = 1.0\,,\quad \sigma_\varepsilon = 1.3\,,\quad Pr_T = 0.9\,. \end{gathered} \tag{7.53}$$

Furthermore, the near-wall damping functions read

$$\begin{aligned} f_\mu &= \exp\left(\frac{-3.4}{(1+0.02\,Re_T)^2}\right) \\ f_{\varepsilon 1} &= 1 \\ f_{\varepsilon 2} &= 1 - 0.3\exp\left(Re_T^2\right) \end{aligned} \tag{7.54}$$

with $Re_T = \rho K^2/(\varepsilon^* \mu_L)$ being the turbulent Reynolds number.

Finally, the explicit wall term ϕ_ε and the value ε_w are defined as

$$\phi_\varepsilon = 2\mu_T \frac{\mu_L}{\rho}\left(\frac{\partial^2 v_s}{\partial y_n^2}\right)^2 \quad \text{and} \quad \varepsilon_w = \frac{2\mu_L}{\rho}\left(\frac{\partial \sqrt{K}}{\partial y_n}\right)^2, \tag{7.55}$$

where v_s stands for the velocity parallel to the wall, and y_n represents the coordinate normal to the wall. In order to avoid an explicit knowledge of wall distance and orientation, it is common to compute the wall term and ε_w from the following Cartesian tensor form [53], [50]

$$\phi_\varepsilon = 2\mu_T \frac{\mu_L}{\rho}\left(\frac{\partial^2 v_i}{\partial x_j \partial x_k}\right)^2 \quad \text{and} \quad \varepsilon_w = \frac{2\mu_L}{\rho}\left(\frac{\partial \sqrt{K}}{\partial x_j}\right)^2 \tag{7.56}$$

instead of Eq. (7.55).

Integral Form

Written in time-dependent integral form for a control volume Ω with a surface element dS, a low Reynolds number K-ε turbulence model reads

$$\frac{\partial}{\partial t}\int_\Omega \vec{W}_T \, d\Omega + \oint_{\partial\Omega} (\vec{F}_{c,T} - \vec{F}_{v,T})\, dS = \int_\Omega \vec{Q}_T \, d\Omega \,. \tag{7.57}$$

The vector of conservative variables takes the form

$$\vec{W}_T = \begin{bmatrix} \rho K \\ \rho\varepsilon^* \end{bmatrix}. \tag{7.58}$$

The vector of convective fluxes is defined

$$\vec{F}_{c,T} = \begin{bmatrix} \rho K V \\ \rho\varepsilon^* V \end{bmatrix}, \tag{7.59}$$

where V denotes the contravariant velocity (see Eq. (2.22)). The vector of viscous fluxes is given by

$$\vec{F}_{v,T} = \begin{bmatrix} n_x \tau_{xx}^K + n_y \tau_{yy}^K + n_z \tau_{zz}^K \\ n_x \tau_{xx}^\varepsilon + n_y \tau_{yy}^\varepsilon + n_z \tau_{zz}^\varepsilon \end{bmatrix} \tag{7.60}$$

with the normal turbulent viscous stresses

$$\tau_{xx}^{K} = \left(\mu_L + \frac{\mu_T}{\sigma_K}\right)\frac{\partial K}{\partial x}, \cdots$$
$$\tau_{xx}^{\varepsilon} = \left(\mu_L + \frac{\mu_T}{\sigma_\varepsilon}\right)\frac{\partial \varepsilon^*}{\partial x}, \cdots \tag{7.61}$$

In Eq. (7.60), n_x, n_y, n_z represent the components of the outward-facing unit normal vector of the surface $\partial\Omega$. The source term is evaluated from

$$\vec{Q}_T = \begin{bmatrix} P - \rho\varepsilon \\ (C_{\varepsilon 1} f_{\varepsilon 1} P - C_{\varepsilon 2} f_{\varepsilon 2}\, \rho\varepsilon^*)\dfrac{\varepsilon^*}{K} + \phi_\varepsilon \end{bmatrix}, \tag{7.62}$$

where P denotes the production term of the turbulent kinetic energy. It is defined as

$$\begin{aligned} P = {} & \tau_{xx}^F \frac{\partial u}{\partial x} + \tau_{yy}^F \frac{\partial v}{\partial y} + \tau_{zz}^F \frac{\partial w}{\partial z} \\ & + \tau_{xy}^F \left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right) + \tau_{xz}^F \left(\frac{\partial u}{\partial z} + \frac{\partial w}{\partial x}\right) + \tau_{yz}^F \left(\frac{\partial v}{\partial z} + \frac{\partial w}{\partial y}\right) \end{aligned} \tag{7.63}$$

with the Favre-averaged turbulent stresses τ_{ij}^F given by Eq. (7.25). The constants, the near-wall damping functions as well as the wall term follow for the Launder-Sharma model from the definitions in Eqs. (7.52)-(7.56). The turbulent eddy viscosity μ_T is obtained from Eq. (7.51).

Initial and Boundary Conditions

The simplest approach is to initialise K and ε^* with their freestream values. A better alternative consists of prescribing profiles for K and ε^* near solid walls. The profiles can be obtained from analogy to turbulent flat-plate boundary layer [44]. However, this requires the knowledge of wall distances which may not be readily available like on unstructured grids.

The proper boundary conditions at solid walls are $K = 0$ and $\varepsilon^* = 0$, provided the transformation in Eq. (7.52) is utilised. This also implies $\mu_T = 0$ at walls. At inflow boundaries, K and ε^* can be computed from relations for the turbulent intensity and length scale, i.e.,

$$(T_u)_\infty = \frac{\sqrt{\frac{2}{3}K_\infty}}{\|\vec{v}_\infty\|_2}, \quad (l_T)_\infty = \frac{C_\mu K_\infty^{3/2}}{\varepsilon_\infty^*}, \tag{7.64}$$

where we assumed $\varepsilon_\infty^* = \varepsilon_\infty$. In turbomachinery, $(l_T)_\infty$ is chosen between 10^{-3} and 10^{-2} times the mean radial blade spacing [54]. The values of K and ε^* are extrapolated from the interior at outflow boundaries.

Wall functions

As we already noted, the low Reynolds number models require very fine grids at walls. The standard condition is that the first node (or cell centroid) should be located at the distance $y^+ \leq 1$ from the wall. In order to reduce the the stiffness of the turbulence equations and to save the number of grid points/cells, coarser grids with $10 \leq y^+ \leq 100$ are sometimes employed. In such a case, the K-ε model Eq. (7.50) or Eq. (7.57) is applied without the damping functions ($f_\mu = f_{\varepsilon 1} = f_{\varepsilon 2} = 1$; $\varepsilon_w = 0$) and the wall term ($\phi_\varepsilon = 0$). We speak here of a *high Reynolds number* turbulence model. Apparently, the distance between the first node (cell centroid) and the wall has to be bridged by the so-called *wall functions*. The wall functions deliver the values of K and ε^* at the node (cell centroid) adjacent to the wall. The turbulence equations are not solved at the wall itself and at the first layer of nodes (cells). Various formulation of the wall functions are used, in general based on the logarithmic wall-law. One example is the function of Spalding [55], which models the viscous sublayer, the transition region as well as the logarithmic layer. Implementations of high Reynolds number models were described, e.g., in [56]-[60] or [51].

The application of the wall functions leads (provided the grid is not too coarse) to reasonably accurate results for attached boundary layers. It also allows the utilisation of purely explicit time-stepping schemes. However, the use of wall functions becomes highly questionable for separated flows.

7.2.3 SST Two-Equation Model of Menter

The K-ω Shear Stress Transport (SST) turbulence model of Menter (Refs. [61], [62]) merges the K-ω model of Wilcox [63], [17] with a high Reynolds number K-ε model (transformed into the K-ω formulation). The SST model seeks to combine the positive features of both models. Therefore, the K-ω approach is employed in the sublayer of the boundary layer. The reason is that the K-ω model needs no damping function. This leads, for similar accuracy, to significantly higher numerical stability in comparison to the K-ε model. Furthermore, the K-ω model is also utilised in logarithmic part of the boundary layer, where it is superior to the K-ε approach in adverse pressure flows and in compressible flows. On the other hand, the K-ε model is employed in the wake region of the boundary layer because the K-ω model is strongly sensitive to the freestream value of ω [64]. The K-ε approach is also used in in free shear layers since it represents a fair compromise in accuracy for wakes, jets, and mixing layers.

One distinct feature of the SST turbulence model is the modified turbulent eddy-viscosity function. The purpose is to improve the accuracy of prediction of flows with strong adverse pressure gradients and of pressure-induced boundary layer separation. The modification accounts for the transport of the turbulent shear stress. It is based on the observation of Bradshaw that the principal shear stress is proportional to the turbulent kinetic energy.

A certain disadvantage of the SST model is that distances to nearest wall

have to be known explicitly. This requires special provisions on multiblock structured or on unstructured grids. Examples for applications of the SST turbulence model can be found in [65]-[67].

Differential Form

The transport equations for the turbulent kinetic energy and the specific dissipation of turbulence read in differential form [61]

$$\frac{\partial \rho K}{\partial t} + \frac{\partial}{\partial x_j}(\rho v_j K) = \frac{\partial}{\partial x_j}\left[(\mu_L + \sigma_K \mu_T)\frac{\partial K}{\partial x_j}\right] + \tau_{ij}^F S_{ij} - \beta^* \rho \omega K$$

$$\frac{\partial \rho \omega}{\partial t} + \frac{\partial}{\partial x_j}(\rho v_j \omega) = \frac{\partial}{\partial x_j}\left[(\mu_L + \sigma_\omega \mu_T)\frac{\partial \omega}{\partial x_j}\right] + \frac{C_\omega \rho}{\mu_T}\tau_{ij}^F S_{ij} \tag{7.65}$$

$$- \beta \rho \omega^2 + 2(1 - f_1)\frac{\rho \sigma_{\omega 2}}{\omega}\frac{\partial K}{\partial x_j}\frac{\partial \omega}{\partial x_j}.$$

The terms on the right-hand side of Eq. (7.65) represent conservative diffusion, eddy-viscosity production and dissipation, respectively. Furthermore, the last term in the ω-equation describes the cross diffusion. The Favre-averaged turbulent stresses τ_{ij}^F are given by Eq. (7.25) and the strain-rate tensor S_{ij} follows from Eq. (7.3). The turbulent eddy viscosity in Eq. (7.28) and (7.29) is obtained from [61]

$$\mu_T = \frac{a_1 \rho K}{\max(a_1 \omega,\ f_2 \|\text{curl}\,\vec{v}\|_2)}. \tag{7.66}$$

This definition of the turbulent viscosity guarantees that in an adverse pressure gradient boundary layer, where the production of K is larger than its dissipation ω (hence $a_1\omega < \|\text{curl}\,\vec{v}\|_2$), Bradshaw's assumption, i.e., $\tau = a_1 \rho K$ (shear stress proportional to turbulent kinetic energy) is satisfied.

The function f_1 in Eq. (7.65), which blends the model coefficients of the K-ω model in boundary layers with the transformed K-ε model in free-shear layers and freestream zones, is defined as

$$f_1 = \tanh(arg_1^4)$$

$$arg_1 = \min\left[\max\left(\frac{\sqrt{K}}{0.09\omega d},\ \frac{500\,\mu_L}{\rho \omega d^2}\right),\ \frac{4\rho\sigma_{\omega 2}K}{CD_{K\omega}d^2}\right], \tag{7.67}$$

where d stands for the distance to the nearest wall and $CD_{K\omega}$ is the positive part of the cross-diffusion term in Eq. (7.65), i.e.,

$$CD_{K\omega} = \max\left(2\frac{\rho\sigma_{\omega 2}}{\omega}\frac{\partial K}{\partial x_j}\frac{\partial \omega}{\partial x_j},\ 10^{-20}\right). \tag{7.68}$$

The auxiliary function f_2 in Eq. (7.66) is given by

$$f_2 = \tanh(arg_2^2)$$
$$arg_2 = \max\left(\frac{2\sqrt{K}}{0.09\omega d}, \frac{500\,\mu_L}{\rho\omega d^2}\right). \tag{7.69}$$

The model constants are as follows

$$a_1 = 0.31\,, \quad \beta^* = 0.09\,, \quad \kappa = 0.41\,. \tag{7.70}$$

Finally, the coefficients of the SST turbulence model β, C_ω, σ_K, and σ_ω are obtained by blending the coefficients of the K-ω model, denoted as ϕ_1, with those of the transformed K-ε model (ϕ_2). The corresponding relation reads

$$\phi = f_1\phi_1 + (1 - f_1)\phi_2\,. \tag{7.71}$$

The coefficients of the inner model (K-ω) are given by

$$\begin{aligned} \sigma_{K1} = 0.85\,, \quad \sigma_{\omega 1} = 0.5\,, \quad \beta_1 = 0.075\,, \\ C_{\omega 1} = \beta_1/\beta^* - \sigma_{\omega 1}\kappa^2/\sqrt{\beta^*} = 0.533\,. \end{aligned} \tag{7.72}$$

The coefficients of the outer model (K-ε) are defined as

$$\begin{aligned} \sigma_{K2} = 1.0\,, \quad \sigma_{\omega 2} = 0.856\,, \quad \beta_2 = 0.0828\,, \\ C_{\omega 2} = \beta_2/\beta^* - \sigma_{\omega 2}\kappa^2/\sqrt{\beta^*} = 0.440\,. \end{aligned} \tag{7.73}$$

The integral formulation of the SST turbulence model corresponds, in principle, to that of the K-ε model from Subsection 7.2.2. Therefore, it is not repeated here.

Boundary Conditions

The boundary conditions for the kinetic turbulent energy and the specific dissipation at solid walls are

$$K = 0 \qquad \text{and} \quad \omega = 10\frac{6\,\mu_L}{\rho\beta_1(d_1)^2} \tag{7.74}$$

with d_1 being the distance of the first node (cell centroid) from the wall. The grid has to be refined such that $y^+ < 3$.

For the inflow boundaries, the following freestream values are recommended

$$\omega_\infty = C_1\frac{\|\vec{v}_\infty\|_2}{L}\,, \quad (\mu_T)_\infty = (\mu_L)_\infty 10^{-C_2}\,, \quad K_\infty = \frac{(\mu_T)_\infty}{\rho_\infty}\omega_\infty\,, \tag{7.75}$$

where L denotes the length of the computational domain, $1 \le C_1 \le 10$ and $2 \le C_2 \le 5$, respectively. The values of K and ω are extrapolated from the interior at outflow boundaries.

7.3 Large-Eddy Simulation

The *Large-Eddy Simulation* (LES) methodology was employed already in 1963 by Smagorinsky in meteorology [68] (circulation of the atmosphere). The first engineering application of LES (turbulent channel flow) was presented by Deardorff in 1970 [69]. His method was later extended and improved by Schumann [70]. During 1980's, the research focus in the simulation of turbulence shifted from LES to Direct Numerical Simulation (DNS). However, some important work was still conducted, e.g., by Bardina et al. [71], Moin and Kim [72]. The interest in LES returned back at the beginning of 1990's [73]-[79]. Nowadays, LES is increasingly employed for physically and geometrically complex flows of engineering relevance like, e.g., in combustion chambers. Examples can be found in Refs. [80]-[91]. Certainly, this trend is supported by the availability of low-cost, highly powerful computers. Additionally, today's engineers are also often faced with flow problems, for which the standard turbulence models fail. Furthermore, in certain cases the mean flow frequencies are in the same order as the turbulent fluctuations. Hence, the time averaging looses its sense and we have to employ either LES or DNS.

LES is based on the observation that the small turbulent structures are more universal in character than the large eddies. Therefore, the idea is to compute the contributions of the large, energy-carrying structures to momentum and energy transfer and to model the effects of the small structures, which are not resolved by the numerical scheme. Due to the more homogeneous and universal character of the small scales, we may expect that the so called *subgrid-scale* models can be kept much simpler than the turbulence models for the RANS equations.

LES represents a 3-D, time-dependent solution of the governing equations. In comparison to turbulence modelling based on the RANS equations, LES requires high grid resolution also in the streamwise ($50 \leq x^+ \leq 150$) and in the cross-flow direction ($15 \leq z^+ \leq 40$). However, LES is computationally considerably cheaper than DNS. The number of grid points (cells) required to resolve the outer layer is proportional to $Re^{0.4}$ [92]. The resolution has to be increased like $Re^{1.8}$ in the viscous sublayer. Thus, if compared to $Re^{9/4}$ required by DNS, LES can be applied at Reynolds numbers at least one order of magnitude higher. In order to further reduce the requirements on grid resolution, LES can be used in conjuction with approximate wall models (Subsection 7.3.4). This approach allows for LES of engineering problems at reasonable computational costs.

An accurate resolution of high wave-number turbulent fluctuations requires spatial discretisation schemes with corresponding properties in the wave-number space (cf., e.g., [93] or [94]). Therefore, spectral methods are often employed. However, spectral methods are applicable only to geometrically simple domains with (quasi-)periodic boundaries. This is the reason why finite difference or finite volume spatial discretisations are becoming increasingly popular. Central differencing schemes proved to be more suitable than upwind schemes. The reason is that upwind schemes (regardless of the order of accuracy) dissipate too much energy over a significant portion of the turbulent spectra due to the inher-

ent numerical damping [95], [96]. A discussion of numerical errors of spectral and finite difference methods can be found in [97].

The implementation of LES methods on unstructured grids [98]-[102] represents a particular challenge. However, it allows for the treatment of highly complex geometries, moving boundaries or for dynamic grid adaptation. The research topic consists of the development of numerically efficient, high-order spatial discretisation on mixed-element grids.

An introduction to LES can be found in Ref. [19], pp. 269-336, and in [103]-[105] or [83]. An overview of the present state of LES was given in [106].

7.3.1 Spatial Filtering

LES is based on a spatial filtering operation, which decomposes any flow variable U into a filtered (large-scale, resolved) part $\overline{U}$ and into a sub-filter (unresolved) part U', i.e.,

$$U = \overline{U} + U' \tag{7.76}$$

The filtered variable at the location $\vec{r}_0$ in space is defined as

$$\overline{U}(\vec{r}_0, t) = \int_D U(\vec{r}, t)\, G(\vec{r}_0, \vec{r}, \Delta)\, d\vec{r}, \tag{7.77}$$

where Ω denotes the entire flow domain, G represents the filter function, and $\vec{r}$ is the position vector, respectively. The filter function determines the structure and size of the small scales. The filter function depends on the difference $\vec{r}_0 - \vec{r}$ and on the filter width $\Delta = (\Delta_1\, \Delta_2\, \Delta_3)^{1/3}$, with Δ_i being the filter width in the i-th spatial coordinate. The following filter functions are the mostly used ones (see Fig. 7.3):

1. the tophat filter:

$$G = \begin{cases} 1/\Delta^3 & \text{if } |(x_0)_i - x_i| \leq \Delta_i/2 \\ 0 & \text{otherwise.} \end{cases} \tag{7.78}$$

2. The sharp Fourier cut-off filter:

$$G = \prod_{i=1}^{3} \frac{\sin\left(\dfrac{\pi}{\Delta_i}[(x_0)_i - x_i]\right)}{\pi\,[(x_0)_i - x_i]} \,. \tag{7.79}$$

3. The Gaussian filter:

$$G = \left(\frac{6}{\pi\Delta^2}\right)^{3/2} \exp\left(\frac{-6\,\|\vec{r}_0 - \vec{r}\|_2^2}{\Delta^2}\right) \,. \tag{7.80}$$

The tophat and the Gaussian filter smooth the large-scale fluctuations as well as the small scales below the filter width. The cut-off filter affects only the scales below the cut-off wave-number. In practice, the Gaussian filter is always employed in conjuction with a sharp Fourier cut-off. Filters suitable for grids with varying cell sizes were proposed in Refs. [107], [108].

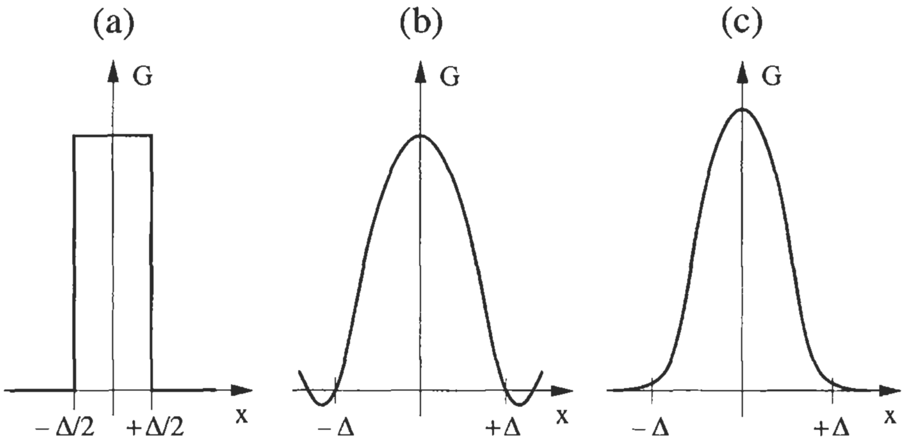


Figure 7.3: LES filter functions in physical space: tophat (a), cut-off (b), Gaussian (c).

7.3.2 Filtered Governing Equations

The spatial filtering, defined by Eq. (7.76) and Eq. (7.77), has to be applied to the Navier-Stokes equations in order to remove the small turbulent scales. The filter width Δ as well as the filter function are considered as free parameters. In fact, the governing equations are usually not explicitly filtered. Instead, the grid as well as the discretisation errors are assumed to define the filter G. For the discussion of explicit filtering see Refs. [109], [110].

Because of the differing treatment, we shall distinguish in the following between compressible (7.1) and incompressible (7.6) formulation of the Navier-Stokes equations.

Incompressible Navier-Stokes Equations

For an incompressible flow of a Newtonian fluid, the filtered governing equations (7.6) take the form

$$\begin{aligned} \frac{\partial \overline{v}_i}{\partial x_i} &= 0 \\ \frac{\partial \overline{v}_i}{\partial t} + \frac{\partial}{\partial x_j}(\overline{v}_i \overline{v}_j) &= -\frac{1}{\rho}\frac{\partial \overline{p}}{\partial x_i} + \nu \nabla^2 \overline{v}_i - \frac{\partial \tau_{ij}^S}{\partial x_j}\,, \end{aligned} \tag{7.81}$$

where ν denotes the kinematic viscosity coefficient. The equations (7.81) describe the temporal and spatial evolution of the large, energy-carrying scales of motion. The non-linearity of the convective term leads to the appearance of the so-called *subgrid-scale stress* (SGS) tensor

$$\tau_{ij}^S = \overline{v_i v_j} - \overline{v}_i \overline{v}_j\,, \tag{7.82}$$

which describes the effects of the unresolved scales. The SGS tensor has to be modelled (see Subsection 7.3.3) in order to close the equations.

The SGS tensor can be decomposed into three parts [111], namely

$$\tau_{ij}^{S} = L_{ij} + C_{ij} + \tau_{ij}^{SR}. \tag{7.83}$$

The individual parts have the following physical meaning:

$$L_{ij} = \overline{\overline{v}_i \overline{v}_j} - \overline{v}_i \overline{v}_j \tag{7.84}$$

is the so-called *Leonard stress* term and represents the interactions between large-scale eddies which produce small-scale turbulence. This term only can be evaluated explicitly from the filtered velocity field v_i. Further, the cross-stress term

$$C_{ij} = \overline{\overline{v}_i v_j'} + \overline{v_i' \overline{v}_j} \tag{7.85}$$

describes interactions between large- and small-scale eddies. Finally,

$$\tau_{ij}^{SR} = \overline{v_i' v_j'} \tag{7.86}$$

is the so-called *SGS Reynolds-stress* tensor. It reflects interactions between the small-scale structures. The above decomposition (7.83) is no longer used mainly because L_{ij} and C_{ij} are not invariant with respect to Galilean transformation (Galilean invariance means that all frames of reference which are translating uniformly with respect to each other are equivalent).

Compressible Navier-Stokes Equations

If LES is to be applied to compressible flows, we have employ Favre averaging (Subsection 7.1.2) together with the spatial filtering to the Equations (7.1). Otherwise, the filtered Navier-Stokes equations would contain products between density and other variables like velocity or temperature. Thus, the velocity components, the energy and the temperature in Eq. (7.1) is decomposed as

$$U = \tilde{U} + U''. \tag{7.87}$$

The filtered variable at the location $\vec{r}_0$ in space is given by

$$\tilde{U}(\vec{r}_0, t) = \frac{\overline{\rho U}}{\overline{\rho}} = \frac{1}{\overline{\rho}} \int_D \rho(\vec{r}, t)\, U(\vec{r}, t)\, G(\vec{r}_0, \vec{r}, \Delta)\, d\vec{r}, \tag{7.88}$$

where the overbar denotes the filtering in Eq. (7.77). The Favre-filtered Navier-Stokes equations (7.1) read [104], [106]

$$\frac{\partial \overline{\rho}}{\partial t} + \frac{\partial}{\partial x_j}(\overline{\rho}\tilde{v}_j) = 0$$

$$\frac{\partial \overline{\rho}\tilde{v}_i}{\partial t} + \frac{\partial (\overline{\rho}\tilde{v}_j\tilde{v}_i)}{\partial x_j} + \frac{\partial \overline{p}}{\partial x_i} - \frac{\partial \hat{\sigma}_{ij}}{\partial x_j} = -\frac{\partial \tau_{ij}^{SF}}{\partial x_j} + \frac{\partial}{\partial x_j}(\overline{\sigma}_{ij} - \hat{\sigma}_{ij}) \tag{7.89}$$

$$\frac{\partial \overline{\rho}\tilde{e}}{\partial t} + \frac{\partial (\overline{\rho}\tilde{v}_j\tilde{e})}{\partial x_j} + \frac{\partial \hat{q}}{\partial x_j} + \overline{p}\tilde{S}_{kk} - \hat{\sigma}_{ij}\tilde{S}_{ij} = -\mathcal{A} - \mathcal{B} - \mathcal{C} + \mathcal{D}$$

with the terms

$$\mathcal{A} = \frac{\partial}{\partial x_j}\left[\overline{\rho}(\widetilde{v_j e} - \tilde{v}_j \tilde{e})\right] \quad \text{– divergence of subgrid-scale heat flux}$$

$$\mathcal{B} = \frac{\partial}{\partial x_j}\left[\overline{q}_j - \hat{q}_j\right] \quad \text{– divergence of SGS heat diffusion}$$

$$\mathcal{C} = \left[\overline{pS_{kk}} - \overline{p}\tilde{S}_{kk}\right] \quad \text{– SGS pressure-dilatation}$$

$$\mathcal{D} = \left[\overline{\sigma_{ij}S_{ij}} - \hat{\sigma}_{ij}\tilde{S}_{ij}\right] \quad \text{– SGS viscous dissipation}$$

and

$$\begin{aligned}
\overline{\sigma}_{ij} &= \overline{2\mu S_{ij}} + \overline{\left(\mu_B - \frac{2\mu}{3}\right)\delta_{ij}S_{kk}} \\
\hat{\sigma}_{ij} &= 2\tilde{\mu}\tilde{S}_{ij} + \left(\tilde{\mu}_B - \frac{2\tilde{\mu}}{3}\right)\delta_{ij}\tilde{S}_{kk} \\
\tilde{S}_{ij} &= \frac{1}{2}\left(\frac{\partial \tilde{v}_i}{\partial x_j} + \frac{\partial \tilde{v}_j}{\partial x_i}\right) \\
\overline{q}_j &= -\overline{k\frac{\partial T}{\partial x_j}}, \quad \hat{q}_j = -\tilde{k}\frac{\partial \tilde{T}}{\partial x_j}.
\end{aligned} \tag{7.90}$$

In the above equations (7.89)-(7.90), e denotes internal energy per unit mass, $\tilde{S}_{ij}$ is the Favre-filtered strain-rate tensor, and $\tau_{ij}^{SF} = \overline{\rho}(\widetilde{v_i v_j} - \tilde{v}_i\tilde{v}_j)$ represents the Favre-averaged subgrid-scale stress. Furthermore, μ, μ_B, and k stand for the molecular viscosity, the bulk viscosity, and for the thermal conductivity, respectively. Finally, $\tilde{\mu}$, $\tilde{\mu}_B$, and $\tilde{k}$ are the respective values at the filtered temperature $\tilde{T}$.

The right-hand side of Eq. (7.89) contains terms which have to be modelled. In the momentum equation, the SGS stresses τ_{ij}^{SF} are approximated, but the second term, i.e., $(\overline{\sigma}_{ij} - \hat{\sigma}_{ij})$ is usually neglected. In the energy equation, term $\mathcal{A}$ can be expressed through the SGS stresses [112], term $\mathcal{B}$ can be neglected, and terms $\mathcal{C}$, $\mathcal{D}$ can be modelled as proposed in [113].

7.3.3 Subgrid-Scale Modelling

The main task of a subgrid-scale model is to simulate energy transfer between the large and the subgrid scales. On the average, the energy is transported from the large scales to the small ones (turbulent cascade process). Therefore, a subgrid-scale model has to provide means of adequate energy dissipation. However, in some instances the energy also flows from small to large scales – a process called *backscatter*. Thus, the model should account for this effect as well. Backscatter models are discussed, e.g., in [114].

Various subgrid-scale models were proposed in the past and the research still continues. The majority of the present models is based on the eddy-viscosity

concept, which is explained next. Furthermore, we shall present the *Smagorinsky* model, which forms the basis of all subgrid-scale models. We shall also briefly discuss the basics of the so-called ***dynamic subgrid-scale*** models. A comparison of six different subgrid-scale models was presented recently in [115].

Eddy-Viscosity Models

These models are able to represent the global dissipative effects of the small scales, but they cannot reproduce the local details of the energy exchange. In the case of incompressible flows, the eddy-viscosity models relate the SGS stresses to the large-scale strain-rate $\overline{S}_{ij}$ as follows

$$\tau_{ij}^S - \frac{\delta_{ij}}{3}\tau_{kk}^S = -2\nu_T \overline{S}_{ij}\,. \tag{7.91}$$

The strain-rate $\overline{S}_{ij}$ is obtained from Eq. (7.3) by using filtered velocity components. The eddy viscosity ν_T is in general evaluated from algebraic relations in order to save numerical costs. The isotropic part of the SGS stresses (τ_{kk}^S) can either be added to the filtered pressure [116], modelled [117] or neglected.

Relation similar to Eq. (7.91) applies also in the case of compressible Navier-Stokes equations. The components of the Favre-averaged SGS stress tensor are approximated as

$$\tau_{ij}^{SF} - \frac{\delta_{ij}}{3}\tau_{kk}^{SF} = -2\overline{\rho}\,\nu_T \tilde{S}_{ij} + \left(\frac{2\overline{\rho}\,\nu_T}{3}\right)\frac{\partial \tilde{v}_k}{\partial x_k}\delta_{ij}\,. \tag{7.92}$$

The components of the strain-rate tensor $\tilde{S}_{ij}$ are given in Eq. (7.90).

Smagorinsky SGS Model

The Smagorinsky model [68] is based on the equilibrium hypothesis which implies that the small scales dissipate entirely and instantaneously all the energy they receive from the large scales. The algebraic model takes the form

$$\nu_T = (C_s\Delta)^2|\overline{S}|\,, \tag{7.93}$$

where $|\overline{S}| = (2\overline{S}_{ij}\overline{S}_{ij})^{1/2}$ is the magnitude of the strain-rate tensor and C_s denotes the *Smagorinsky constant.* The theoretical value found by Lilly [118] is $C_s = 0.18$. However, the Smagorinsky constant depends on the type of the flow. For example, in shear flows C_s has to be reduced to approximately 0.1. The filter width Δ in Eq. (7.93) is usually chosen to be twice the average grid size, i.e., $\Delta = 2\,(\Delta x_1\,\Delta x_2\,\Delta x_3)^{1/3}$.

In order to account for the reduced growth of the small scales near walls, the value of the eddy viscosity ν_T has to be reduced. Thus, the Smagorinsky model Eq. (7.93) is modified according to Van Driest damping as

$$\nu_T = \left[C_s\Delta\,(1 - e^{-y^+/25})\right]^2 |\overline{S}|\,, \tag{7.94}$$

where y^+ represents the dimensionless wall distance.

The Smagorinsky model is numerically cheap and easy to implement. However, it has several serious disadvantages:

- it is too dissipative in laminar regions with mean shear;
- it requires special provisions near walls and at laminar-turbulent transition;
- the parameter C_s is not uniquely defined;
- the process of energy backscatter is not modelled.

Because of these shortcomings, various other approaches were proposed (see, e.g., [104]). Very popular are the dynamic models.

Dynamic SGS Models

The dynamic SGS models employ the same relation as Smagorinsky (Eq. (7.93) for the evaluation of the eddy viscosity ν_T in Eq. (7.91) or (7.92). The difference is that the Smagorinsky constant (adjusted a priori) is replaced by a parameter, which evolves dynamically in space and time. Hence,

$$\nu_T = C_d(\vec{r}, t)\, \Delta^2 |\overline{S}| \,. \tag{7.95}$$

The parameter C_d is computed based on the energy content of the smallest scale of turbulence. For this purpose, Germano et al. [119] proposed to employ a second filter – the so-called *test* filter $\hat{\Delta}$. The width of the test filter has to be larger than that of the filter Δ applied to the governing equations (usually $\hat{\Delta} = 2\Delta$). The application of the test filter to the filtered equations leads to the so-called *subtest-scale stresses* τ_{ij}^{ST}

$$\tau_{ij}^{ST} = \widehat{\overline{v_i v_j}} - \hat{\overline{v}}_i \hat{\overline{v}}_j \,. \tag{7.96}$$

The subtest-scale stresses are related to the SGS stresses τ_{ij}^{S} (Eq. (7.83)) via the *Germano identity* [120]

$$\hat{L}_{ij} = \tau_{ij}^{ST} - \hat{\tau}_{ij}^{S} = \widehat{\overline{v}_i \overline{v}_j} - \hat{\overline{v}}_i \hat{\overline{v}}_j \,, \tag{7.97}$$

where $\hat{L}_{ij}$ denotes the Leonard stresses associated with the test filter. It represents the contribution to the Reynolds stresses by the scales whose length is intermediate between the filter width Δ and the test filter width $\hat{\Delta}$.

If we express the subtest-scale and SGS stresses in Eq. (7.97) using the eddy-viscosity approach Eq. (7.91) together with Eq. (7.95), we obtain

$$\hat{L}_{ij} - \frac{\delta_{ij}}{3} L_{kk} = -2 C_d\, M_{ij} \tag{7.98}$$

with

$$M_{ij} = \hat{\Delta}^2 |\hat{\overline{S}}| \hat{\overline{S}}_{ij} - \left[\Delta^2 |\overline{S}| \overline{S}_{ij}\right]^{\wedge} . \tag{7.99}$$

The notation $[]^\wedge$ means that the whole term enclosed in the brackets is test-filtered. The parameter C_d can be derived from Eq. (7.98) by using the least-squares minimisation of Lilly [121]. This leads to

$$C_d(\vec{r}, t) = -\frac{1}{2} \frac{L_{ij} M_{ij}}{M_{mn} M_{mn}}. \tag{7.100}$$

The above formulation (7.100) is mathematically inconsistent since the parameter C_d was taken outside the test filter in Eq. (7.98). In practice, the numerator and denominator in Eq. (7.100) are therefore ensemble-averaged in the homogeneous directions, i.e.,

$$C_d(\vec{r}, t) = -\frac{1}{2} \frac{\langle L_{ij} M_{ij} \rangle}{\langle M_{mn} M_{mn} \rangle}. \tag{7.101}$$

Improved dynamic SGS models were proposed, e.g., by Ghosal et al. [122], Carati et al. [123], Piomelli and Liu [124], and Held [83].

7.3.4 Wall Models

The computational costs of LES of wall-bounded flows at high Reynolds numbers ($Re > 10^6$) are still too high for engineering purposes. The reason is the excessively large number of grid points (cells) required to resolve the wall layer appropriately. In order to reduce the costs, it is possible to model the wall layer by specifying a correlation between the velocity in the outer flow and the stress at the wall. This approach is quite similar to using wall functions in RANS simulations. The basic assumption is that there is only a weak interaction between the near-wall and the outer region, which is supported by the investigations in [125] and [126].

Earlier implementations of the wall models were based on the assumption that the dynamics of the wall layer are universal and hence they can be approximated by a generalised law-of-the-wall. Basically, the models utilised the logarithmic law (see [70] and [127]-[129]). Balaras et al. [130] proposed recently a new zonal approach. Within the two-layer model, the filtered Navier-Stokes equations (7.81) are solved up to the first grid point above the wall. From this point to the wall 2-D boundary layer equation are solved on a refined embedded grid. The solution on the embedded grid is then used to prescribe the wall shear stress as a boundary condition for the LES. The zonal approach of Balaras et al. [130] allows it to place the first point in a region $20 < y^+ < 100$, which leads to significantly reduced grid size and hence computational time. The methodology was applied with success to turbulent flows in a plane channel, square duct and rotating channel. Later on, it was also employed for the LES of separated flows with encouraging results [131], [132].

Bibliography

[1] Liu, C.; Liu, Z.: *Multigrid Methods and High Order Finite Difference for Flow in Transition.* AIAA Paper 93-3354, 1993.

[2] Olejniczak, D.J.; Weirs, V.G.; Liu, J.; Candler, G.V.: *Hybrid Finite-Difference Methods for DNS of Compressible Turbulent Boundary Layers.* AIAA Paper 96-2086, 1996.

[3] Cook, A.W.; Riley, J.J.: *Direct Numerical Simulation of a Turbulent Reactive Plume on a Parallel Computer.* J. Computational Physics, 129 (1996), pp. 263-283.

[4] Pinelli, A.; Vacca, A.; Quarteroni, A.: *A Spectral Multidomain Method for the Numerical Simulation of Turbulent Flows.* J. Computational Physics, 136 (1997), pp. 546-558.

[5] Zhong, X.: *High-Order Finite-Difference Schemes for Numerical Simulation of Hypersonic Boundary-Layer Transition.* J. Computational Physics, 144 (1998), pp. 662-709.

[6] Lange, M.; Riedel, U.; Warnatz, J.: *Parallel DNS of Turbulent Flames with Detailed Reaction Schemes.* AIAA Paper 98-2979, 1998.

[7] Hernandez, G.; Brenner, G.: *Boundary Conditions for Direct Numerical Simulations of Free Jets.* AIAA Paper 99-0287, 1999.

[8] Ladeinde, F.; Liu, W.; O'Brien, E.E.: *DNS Evaluation of Chemistry Models for Turbulent, Reacting, and Compressible Mixing Layers.* AIAA Paper 99-0413, 1999.

[9] Freund, J.B.: *Acoustic Sources in a Turbulent Jet - A Direct Numerical Simulation Study.* AIAA Paper 99-1858, 1999.

[10] Rizzetta, D.P.; Visbal, M.R.: *Application of a High-order Compact Difference Scheme to Large-Eddy and Direct Numerical Simulation.* AIAA Paper 99-3714, 1999.

[11] Boussinesq, J.: *Essai sur la théorie des eaux courantes.* Mem. Pres. Acad. Sci., XXIII, 46, Paris, 1877.

[12] Boussinesq, J.: *Theorie de l' écoulement tourbillonant et tumulteur des liquides dans les lits rectiligues.* Comptes Rendus de l' Acad. des Sciences, CXXII (1896), p. 1293.

[13] Reynolds, O.: *On the Dynamical Theory of Incompressible Viscous Fluids and the Determination of the Criterion.* Phil. Trans. Roy. Soc., London, Series A 186 (1895), pp. 123-164.

[14] Favre, A.: *Equations des gaz turbulents compressibles, part 1: formes générales.* Journal de Mécanique 4 (1965), pp. 361-390.

[15] Favre, A.: *Equations des gaz turbulents compressibles, part 2: méthode des vitesses moyennes; methode des vitesses moyennes pondérées par la masse volumique.* Journal de Mécanique 4 (1965), pp. 391-421.

[16] Morkovin, M.V.: *Effects of Compressibility on Turbulent Flow.* The Mechanics of Turbulence, Favre, A. (ed.), Gordon and Breach, New York, 1964.

[17] Wilcox, D.C.: *Turbulence Modeling for CFD.* Published by DCW Industries, Inc., La Cañada, California, USA, 1993.

[18] Reynolds, O.: *On the Extent and Action of the Heating Surface for Steam Boilers.* Proc. Manchester Lit. Phil. Soc., 14 (1874), pp. 7-12.

[19] Hallbäck, M.; Henningson, D.S.; Johansson, A.V.; Alfredsson, P.H. (eds.): *Turbulence and Transition Modelling.* ERCOFTAC Series, Vol. 2, Kluwer Academic Publishers, Dordrecht, The Netherlands, 1996.

[20] Spalart, P.; Shur, M.: *On the Sensitization of Turbulence Models to Rotation and Curvature.* Aerospace Science and Technology, 5 (1997), pp. 297-302.

[21] Shur, M.; Strelets, M.; Travin, A.; Spalart, P.: *Turbulence Modeling in Rotating and Curved Channels: Assessment of the Spalart-Shur Correction Term.* AIAA Paper 98-0325, 1998.

[22] Lumley, J.L.: *Toward a Turbulent Constitutive Equation.* J. Fluid Mechanics, 41 (1970), pp. 413-434.

[23] Lumley, J.L.: *Computational Modeling of Turbulent Flows.* Advances in Applied Mechanics, 18 (1978), pp. 123-176.

[24] Craft, T.J.; Launder, B.E.; Suga, K.: *A Non-Linear Eddy-Viscosity Model Including Sensitivity to Stress Anisotropy.* Proc. 10th Symp. Turbulent Shear Flows, Pennsylvania State University, Paper 23:19, 1995.

[25] Shih, T.H.; Zhu, J.; Liou, W.W.; Chen, K.-H.; Liu, N.-S.; Lumley, J.: *Modeling of Turbulent Swirling Flows.* Proc. 11th Symposium on Turbulent Shear Flows, Grenoble, France, 1997; also NASA TM-113112, 1997.

[26] Goldberg, U.; Peroomian, O.; Chakravarthy, S.: *Application of the k-ε-R Turbulence Model to Wall-bounded Compressive Flows.* AIAA Paper 98-0323, 1998.

[27] Abdel Gawad, A.F.; Abdel Latif, O.E.; Ragab, S.A.; Shabaka, I.M.: *Turbulent Flow and Heat Transfer in Rotating Non-Circular Ducts with Non-linear k-ε Model.* AIAA Paper 98-0326, 1998.

[28] Chen, K.-H.; Liu, N.-S.: *Evaluation of a Non-Linear Turbulence Model Using Mixed Volume Unstructured Grids.* AIAA Paper 98-0233, 1998.

[29] Hinze, J.O.: *Turbulence.* 2nd edition, McGraw-Hill, New York, 1975.

[30] Bai, X.S.: *On the Modeling of Turbulent Combustion at Low Mach Numbers.* PhD Thesis, Royal Inst. of Technology, Dept. of Mechanics/Applied CFD, Stockholm, Sweden, 1994.

[31] Adumitroaie, V.; Ristorcelli, J.R.; Taulbee, D.B.: *Progress in Favre-Reynolds Stress Closures for Compressible Flows.* ICASE Report No. 98-21, 1998.

[32] Bardina, J.E.; Huang, P.G.; Coakley, T.J.: *Turbulence Modeling Validation, Testing, and Development.* NASA TM-110446, 1997.

[33] Spalart, S.R.; Allmaras, S.A.: *A One-Equation Turbulence Model for Aerodynamic Flows.* AIAA Paper 92-0439, 1992; also La Recherche Aerospatiale, 1 (1994), pp. 5-21.

[34] Ashford, G.A.: *An Unstructured Grid Generation and Adaptive Solution Technique for High-Reynolds-Number Compressible Flows.* PhD Thesis, The University of Michigan, Dept. of Aerospace Engineering, An Arbor, USA, 1996.

[35] Chou, P.Y.: *On Velocity Correlations and the Solutions of the Equations of Turbulent Fluctuations.* Quart. of Appl. Math., 3 (1945), pp. 38-54.

[36] Jones, W.P.; Launder, B.E.: *The Prediction of Laminarization with a Two-Equation Model of Turbulence.* Int. Journal of Heat and Mass Transfer, 15 (1972), pp. 301-314.

[37] Jones, W.P.; Launder, B.E.: *The Prediction of Low-Reynolds-Number Phenomena with a Two-Equation Model of Turbulence.* Int. Journal of Heat and Mass Transfer, 16 (1973), pp. 1119-1130.

[38] Launder, B.E.; Sharma, B.I.: *Application of the Energy Dissipation Model of Turbulence to the Calculation of Flow Near a Spinning Disc.* Letters in Heat and Mass Transfer, 1 (1974), pp. 131-138.

[39] Launder, B.E.; Spalding, B.: *The Numerical Computation of Turbulent Flows.* Comput. Methods Appl. Mech. Eng., 3 (1974), pp. 269-289.

[40] Lam, C.K.G.; Bremhorst, K.A.: *Modified Form of k-ε Model for Predicting Wall Turbulence.* ASME, J. of Fluid Engineering, 103 (1981), pp. 456-460.

[41] Chien, K.-Y.: *Predictions of Channel and Boundary-Layer Flows with a Low-Reynolds-Number Turbulence Model.* AIAA Journal, 20 (1982), pp. 33-38.

[42] Patel, V.C.; Rodi, W.; Scheurer, G.: *Turbulence Models for Near-Wall and Low Reynolds Number Flows: A Review.* AIAA Journal, 23 (1985), pp. 1308-1319.

[43] Kunz, R.F.; Lakshminarayana, B.: *Stability of Explicit Navier-Stokes Procedures Using k-ε and k-ε/Algebraic Reynolds Stress Turbulence Models.* J. Computational Physics, 103 (1992), pp. 141-159.

[44] Gerolymos, G.A.: *Implicit Multiple-Grid Solution of the Compressible Navier-Stokes Equations Using k-ε Turbulence Closure.* AIAA Journal, 28 (1990), pp. 1707-1717.

[45] Mavriplis, D.J.; Martinelli, L.: *Multigrid Solution of Compressible Turbulent Flow on Unstructured Meshes Using a Two-Equation Model.* AIAA Paper 91-0237, 1991.

[46] Turner, M.G.; Jennions, I.K.: *An Investigation of Turbulence Modeling in Transonic Fans Including a Novel Implementation of an Implicit k-ε Turbulence Model.* J. of Turbomachinery, 115 (1993), pp. 249-260.

[47] Grasso, F.; Falconi, D.: *On High Speed Turbulence Modeling of Shock-Wave Boundary-Layer Interaction.* AIAA Paper 93-0778, 1993.

[48] Koobus, B.: *An Implicit Method for Turbulent Boundary Layers Simulation.* INRIA Report No. 2450, December 1994.

[49] Sondak, D.L.: *Parallel Implementation of the k-ε Turbulence Model.* AIAA Paper 94-0758, 1994.

[50] Gerolymos, G.A.; Vallet, I.: *Implicit Computation of Three-Dimensional Compressible Navier-Stokes Equations Using k-ε Closure.* AIAA Journal, 34 (1996), pp. 1321-1330.

[51] Luo, H.; Baum, J.D.; Löhner, R.: *Computation of Compressible Flows Using a Two-Equation Turbulence Model on Unstructured Grids.* AIAA Paper 97-0430, 1997.

[52] Wang, Q.; Massey, S.J.; Abdol-Hamid, K.S.; Frink, N.T.: *Solving Navier-Stokes Equations with Advanced Turbulence Models on Three-Dimensional Unstructured Grids.* AIAA Paper 99-0156, 1999.

[53] Lakshminarayana, B.: *Turbulence Modeling for Complex Shear Flows.* AIAA Journal, 24 (1986), pp. 1900-1917.

[54] Kunz, R.F.; Lakshminarayana, B.: *Explicit Navier-Stokes Computation of Cascade Flows Using the k-ε Turbulence Model.* AIAA Journal, 30 (1992), pp. 13-22.

[55] Spalding, D.B.: *A Single Formula for the "Law of the Wall".* ASME J. Applied Mechanics, No. 28, Sept. 1961, pp. 455-458.

[56] Viegas, J.R.; Rubesin, M.W.: *Wall-Function Boundary Conditions in the Solution of the Navier-Stokes Equations for Complex Compressible Flows.* AIAA Paper 83-1694, 1983.

[57] Abdol-Hamid, K.S.; Lashmanan, B.; Carlson, J.R.: *Application of Navier-Stokes Code PAB3D with k-ε Turbulence Models to Attached and Separated Flows.* NASA TR-3480, 1995.

[58] Frink, N.: *Assessment of an Unstructured-Grid Method for Predicting 3-D Turbulent Viscous Flows.* AIAA Paper 96-0292, 1996.

[59] Mohammadi, B.; Pironneau, O.: *Unsteady Separated Turbulent Flows Computation with Wall-Laws and k-ε Model.* Comput. Methods Appl. Engrg., 148 (1997), pp. 393-405.

[60] Grotjans, H.; Menter, F.R.: *Wall Functions for General Application CFD Codes.* Proc. Fourth European CFD Conf., 7-11 Sept. 1998, Athens, Greece.

[61] Menter, F.R.: *Two-Equation Eddy-Viscosity Turbulence Models for Engineering Applications.* AIAA Paper 93-2906, 1993; also AIAA Journal, 32 (1994), pp. 1598-1605.

[62] Menter, F.R.; Rumsey, L.C.: *Assessment of Two-Equation Turbulence Models for Transonic Flows.* AIAA Paper 94-2343, 1994.

[63] Wilcox, D.C.: *Reassessment of the Scale-Determining Equation for Advanced Turbulence Models.* AIAA Journal, 26 (1988), pp. 1299-1310.

[64] Menter, F.R.: *Influence of Freestream Values on k-ω Turbulence Model Predictions.* AIAA Journal, 30 (1992), pp. 1651-1659.

[65] Hellsten, A.; Laine, S.: *Extension of the k-ω-SST Turbulence Model for Flows over Rough Surfaces.* AIAA Paper 97-3577, 1997; also AIAA Journal, 36 (1998), pp. 1728-1729.

[66] Hellsten, A.: *Some Improvements in Menter's k-ω SST Turbulence Model.* AIAA Paper 98-2554, 1998.

[67] Forsythe, J.R.; Hoffmann, K.A.; Suzen, Y.B.: *Investigation of Modified Menter's Two-Equation Turbulence Model for Supersonic Applications.* AIAA Paper 99-0873, 1999.

[68] Smagorinsky, J.: *General Circulation Experiments with the Primitive Equations.* Monthly Weather Review, 91 (1963), pp. 99-165.

[69] Deardorff, J.W.: *A Numerical Study of Three-Dimensional Turbulent Channel Flow at Large Reynolds Numbers.* J. Fluid Mechanics, 41 (1970), pp. 453-480.

[70] Schumann, U.: *Subgrid-Scale Model for Finite-Difference Simulations of Turbulent Flows in Plane Channels and Annuli.* J. Computational Physics, 18 (1975), pp. 376-404.

[71] Bardina, J.; Ferziger, J.H.; Reynolds, W.C.: *Improved Subgrid Models for Large Eddy Simulation.* AIAA Paper 80-1357, 1980.

[72] Moin, P.; Kim, J.: *Numerical Investigation of Turbulent Channel Flow.* J. Fluid Mechanics, 118 (1982), pp. 341-377.

[73] Piomelli, U.; Zang, T.A.; Speziale, C.G., Hussaini, M.Y.: *On the Large-Eddy Simulation of Transitional Wall-Bounded Flows.* ICASE Report No. 89-55, 1989.

[74] Hussaini, M.Y.; Speziale, C.G.; Zang, T.A.: *Discussion of "The Potential and Limitations of Direct and Large-Eddy Simulations".* ICASE Report No. 89-61, 1989.

[75] Erlebacher, G.; Hussaini, M.Y.; Speziale, C.G.; Zang, T.A.: *Toward the Large-Eddy Simulation of Compressible Turbulent Flows.* ICASE Report No. 90-76, 1990.

[76] Erlebacher, G.; Hussaini, M.Y.; Speziale, C.G.; Zang, T.A.: *On the Large-Eddy Simulation of Compressible Isotropic Turbulence.* 12th International Conf. on Numerical Methods in Fluid Dynamics, Oxford, England, Lecture Notes in Physics, Vol. 371, Springer, 1990.

[77] Schumann, U.: *Direct and Large-Eddy Simulation of Turbulence – Summary of the State of the Art 1993.* VKI Lecture Series 1993-02, 1993.

[78] Jones, W.P.; Wille, M.: *Large Eddy Simulation of a Jet in a Cross-Flow.* 10th Symposium on Turbulent Shear Flows, The Pennsylvania State University, PA, August 14-16, 1995, Volume 1.

[79] Lund, T.S.; Moin, P.: *Large Eddy Simulation of a Boundary Layer on a Concave Surface.* 10th Symposium on Turbulent Shear Flows, The Pennsylvania State University, PA, August 14-16, 1995, Volume 1.

[80] Calhoon, W.H.; Menon, S.: *Subgrid Modeling for Reacting Large Eddy Simulations.* AIAA Paper 96-0516, 1996.

[81] Cook, A.W.; Riley, J.J.; Kosaly, G.: *A Laminar Flamelet Approach to Subgrid-Scale Chemistry in Turbulent Flows.* Combustion and Flame, 109 (1997), pp. 332-341.

[82] Wu, Y.; Cao, S.; Yang, J.; Ge, L.; Guilbaud, M.; Soula, V.: *Turbulent Flow Calculation Through a Water Turbine Draft Tube by Using the Smagorinsky Model.* ASME Fluids Engineering Division Summer Meeting (FEDSM), Washington D.C., Paper No. FEDSM98-4864, 1998.

[83] Held, J.: *Large Eddy Simulations of Separated Compressible Flows Around 3-D Configurations.* PhD Thesis, Department of Heat and Power Engineering/Fluid Mechanics, Lund Institute of Technology, Lund, Sweden, 1998.

[84] Kim, W.-W.; Menon, S.; Mongia, H.C.: *Large Eddy Simulations of Reacting Flows in a Dump Combustor.* AIAA Paper 98-2432, 1998.

[85] Ducros, F.; Ferrand, V.; Nicoud, F.; Weber, C.; Darracq, D.; Gacherieu, C.; Poinsot, T.: *Large-Eddy Simulation of the Shock/Turbulence Interaction.* J. Computational Physics, 152 (1999), pp. 517-549.

[86] Jaberi, F.A.: *Large Eddy Simulation of Turbulent Premixed Flames via Filtered Mass Density Function.* AIAA Paper 99-0199, 1999.

[87] Han, J.; Lin, Y.-L.; Arya, S.P.; Proctor, F.H.: *Large Eddy Simulation of Aircraft Wake Vortices in a Homogeneous Atmospheric Turbulence – Vortex Decay and Descent.* AIAA Paper 99-0756, 1999.

[88] Bui, T.T.: *A Parallel, Finite-Volume Algorithm for Large-Eddy Simulation of Turbulent Flows.* AIAA Paper 99-0789, 1999.

[89] Kravchenko, A.G.; Moin, P.; Shariff, K.: *B-Spline Method and Zonal Grids for Simulations of Complex Turbulent Flows.* J. Computational Physics, 151 (1999), pp. 757-789.

[90] Cook, A.W.: *A Consistent Approach to Large Eddy Simulation Using Adaptive Mesh Refinement.* J. Computational Physics, 154 (1999), pp. 117-133.

[91] Pascarelli, A.; Piomelli, U.; Candler, G.V.: *Multi-Block Large-Eddy Simulations of Turbulent Boundary Layers.* J. Computational Physics, 157 (2000), pp. 256-279.

[92] Chapman, D.R.: *Computational Aerodynamics Development and Outlook.* AIAA Journal, 17 (1979), pp. 1293-1313.

[93] Tam, C.K.W.; Webb, J.C.: *Dispersion-Relation-Preserving Finite Difference Schemes for Computational Acoustics.* J. Computational Physics, 107 (1993), pp. 262-281.

[94] Tam, C.K.W.: *Applied Aero-Acoustics: Prediction Methods.* VKI Lecture Series 1996-04, 1996.

[95] Mittal, R.; Moin, P.: *Suitability of Upwind-Biased Finite Difference Schemes for Large-Eddy Simulation of Turbulent Flows.* AIAA Journal, 35 (1997), pp. 1415-1417.

[96] Garnier, E.; Mossi, M.; Sagaut, P.; Comte, P.; Deville, M.: *On the Use of Shock-Capturing Schemes for Large-Eddy Simulation.* J. Computational Physics, 153 (1999), pp. 273-311.

[97] Kravchenko, A.G.; Moin, P.: *On the Effect of Numerical Errors in Large Eddy Simulations of Turbulent Flow.* J. Computational Physics, 131 (1997), pp. 310-322.

[98] Miet, P.; Laurence, D.; Nitrosso, B.: *Large Eddy Simulation with Unstructured Grids and Finite Elements.* 10th Symposium on Turbulent Shear Flows, The Pennsylvania State University, PA, August 14-16, 1995, Volume 2.

[99] Jansen, K.: *Large-Eddy Simulation of Flow Around a NACA 4412 Airfoil Using Unstructured Grids.* Center for Turbulence Research, NASA Ames/Stanford University, Annual Research Briefs, 1996, pp. 225-232.

[100] Chalot, F.; Marquez, B.; Ravachol, M.; Ducros, F.; Nicoud, F.; Poinsot, T.: *A Consistent Finite Element Approach to Large Eddy Simulation.* AIAA Paper 98-2652, 1998.

[101] Okong'o, N.; Knight, D.: *Compressible Large Eddy Simulation Using Unstructured Grids: Channel and Boundary Layer Flows.* AIAA Paper 98-3315, 1998.

[102] Urbin, G.; Knight, D.; Zheltovodov, A.A.: *Compressible Large Eddy Simulation Using Unstructured Grid: Supersonic Turbulent Boundary Layer and Compression Corner.* AIAA Paper 99-0427, 1999.

[103] Ferziger, J.H.: *Direct and Large Eddy Simulation of Turbulence.* In "Numerical Methods in Fluid Mechanics", A. Vincent (ed.), Centre de Recherches Mathématiques Université de Montréal, Proceedings and Lecture Notes, 16 (1998), pp. 53-97.

[104] Piomelli, U.: *Large-Eddy Simulation of Turbulent Flows.* VKI Lecture Series 1998-05, 1998.

[105] Mossi, M.: *Simulation of Benchmark and Industrial Unsteady Compressible Turbulent Fluid Flows.* PhD Thesis, No. 1958, École Polytechnique Fédérale de Lausanne, Département de Génie Mécanique, Switzerland, 1999.

[106] Piomelli, U.: *Large-Eddy Simulation: Present State and Future Directions.* AIAA Paper 98-0534, 1998.

[107] Vasilyev, O.V.; Lund, T.S.: *A General Theory of Discrete Filtering for LES in Complex Geometry.* Annual Research Briefs, Center for Turbulence Research, NASA Ames/Stanford Univ., 1997, pp. 67-82.

[108] Vasilyev, O.V.; Lund, T.S.; Moin, P.: *A General Class of Commutative Filters for LES in Complex Geometries.* J. Computational Physics, 146 (1998), pp. 82-104.

[109] Lund, T.S.; Kaltenbach, H.-J.: *Experiments with Explicit Filtering for LES Using a Finite-Difference Method.* Annual Research Briefs, Center for Turbulence Research, NASA Ames/Stanford Univ., 1995, pp. 91-105.

[110] Lund, T.S.: *On the Use of Discrete Filters for Large Eddy Simulation.* Annual Research Briefs, Center for Turbulence Research, NASA Ames/Stanford Univ., 1997, pp. 83-95.

[111] Leonard, A.: *Energy Cascade in Large-Eddy Simulations of Turbulent Fluid Flows.* Advances in Geophysics, 18 (1974), pp. 237-248.

[112] Moin, P.; Squires, K.D.; Cabot, W.H.; Lee, S.: *A Dynamic Subgrid-Scale Model for Compressible Turbulence and Scalar Transport.* Physics of Fluids, 3 (1991), pp. 2746-2757.

[113] Vreman, B.; Geurts, B.; Kuerten, H.; Broeze, J.; Wasistho, B.; Streng, M.: *Dynamic Subgrid-Scale Models for LES of Transitional and Turbulent Compressible Flow id 3-D Shear Layers.* 10th Symposium on Turbulent Shear Flows, The Pennsylvania State University, PA, August 14-16, 1995, Volume 1.

[114] Domaradzki, J.A.; Saiki, E.M.: *Backscatter Models for Large-Eddy Simulations.* Theoret. Comput. Fluid Dynamics, 9 (1997), pp. 75-83.

[115] Lenormand, E.; Sagaut, P.; Phuoc, L.T.; Comte, P.: *Subgrid-Scale Models for Large-Eddy Simulations of Compressible Wall Bounded Flows.* AIAA Journal, 38 (2000), pp. 1340-1350.

[116] Rogallo, R.S.; Moin, P.: *Numerical Simulation of Turbulent Flows.* Annu. Rev. Fluid Mech., 16 (1984), pp. 99-137.

[117] Vreman, A.W.: *Direct and Large-Eddy Simulation of the Compressible Turbulent Mixing Layer.* PhD Thesis, Dept. of Applied Mathematics, University of Twente Enschede, The Netherlands, 1995.

[118] Lilly, D.K.: *The Representation of Small-Scale Turbulence in Numerical Simulation Experiments.* Proc. IBM Sci. Comp. Symp. on Environmental Sciences, New York, November 14-16, 1967.

[119] Germano, M.; Piomelli, U.; Moin, P.; Cabot, W.H.: *A Dynamic Subgrid-Scale Eddy Viscosity Model.* Physics of Fluids, 3 (1991), pp. 1760-1765.

[120] Germano, M.: *Turbulence: The Filtering Approach.* J. Fluid Mechanics, 238 (1992), pp. 325-336.

[121] Lilly, D.K.: *A Proposed Modification of the Germano Subgrid-Scale Closure Method.* Physics of Fluids, 4 (1992), pp. 633-635.

[122] Ghosal, S.; Lund, T.S.; Moin, P.; Akselvoll, K.: *A Dynamic Localization Model for Large-Eddy Simulation of Turbulent Flows.* J. Fluid Mechanics, 286 (1995), pp. 229-255.

[123] Carati, D.; Ghosal, S.; Moin, P.: *On the Representation of Backscatter in Dynamic Localization Models.* Physics of Fluids, 7 (1995), p. 606.

[124] Piomelli, U.; Liu, J.: *Large-Eddy Simulation of Rotating Channel Flows Using a Localized Dynamic Model.* Physics of Fluids, 7 (1995), pp. 839-848.

[125] Chapman, D.R.; Kuhn, G.D.: *The Limiting Behavior of Turbulence Near a Wall.* J. Fluid Mechanics, 70 (1986), pp. 265-292.

[126] Brooke, J.W.; Hanratty, T.J.: *Origin of Turbulence-Producing Eddies in a Channel Flow.* Physics of Fluids, 5 (1993), pp. 1011-1022.

[127] Grötzbach, G.: *Direct Numerical and Large Eddy Simulation of Turbulent Channel Flows.* Encyclopedia of Fluid Mechanics, Cheremisinoff, N.P. (ed.), Gulf Publishing, West Orange, 1987, p. 1337.

[128] Mason, P.J.; Callen, N.S.: *On the Magnitude of the Subgrid-Scale Eddy Coefficient in Large-Eddy Simulations of Turbulent Channel Flow.* J. Fluid Mechanics, 162 (1986), pp. 439-462.

[129] Piomelli, U.; Ferziger, J.: Moin, P.; Kim, J.: *New Approximate Boundary Conditions for Large Eddy Simulations of Wall-Bounded Flows.* Physics of Fluids, 1 (1989), pp. 1061-1068.

[130] Balaras, E.; Benocci, C.; Piomelli, U.: *Two-Layer Approximate Boundary Condition for Large-Eddy Simulations.* AIAA Journal, 34 (1996), pp. 1111-1119.

[131] Cabot, W.: *Large-Eddy Simulations with Wall Models.* Annual Research Briefs, Center for Turbulence Research, NASA Ames/Stanford Univ., 1995, pp. 41-50.

[132] Cabot, W.: *Near-Wall Models in Large Eddy Simulations of Flow Behind a Backward-Facing Step.* Annual Research Briefs, Center for Turbulence Research, NASA Ames/Stanford Univ., 1996, pp. 199-210.

Chapter 8

Boundary Conditions

Any numerical simulation can consider only a part of the real physical domain or system. The truncation of the domain leads to artificial boundaries, where we have to prescribe values of certain physical quantities. Furthermore, walls which are exposed to the flow represent natural boundaries of the the physical domain. The numerical treatment of the boundary conditions requires a particular care. An improper implementation can result in inaccurate simulation of the real system. Additionally, the stability and the convergence speed of the solution scheme can be negatively influenced.

The following types of boundary conditions are in general encountered in the numerical solution of the Euler and the Navier-Stokes equations:

- solid wall,
- farfield in external and inflow/outflow in internal flows,
- symmetry,
- coordinate cut and periodic boundary,
- boundary between blocks.

The numerical treatment of these boundary conditions is described in detail in the following sections. For literature on further boundary conditions like heat radiation on walls or like free surfaces, the reader is referred to Section 3.4.

8.1 Concept of Dummy Cells

Before we proceed with the discussion of the boundary conditions, we should mention the concept of *dummy cells* (also known as *dummy points*). This approach is very popular on structured grids. However, dummy cells offer some advantages also on unstructured grids. The dummy cells are additional layers of grid points outside the physical domain. This is sketched in Fig. 8.1 for the case of a 2-D structured grid. As we can see, the whole computational domain is surrounded by two layers of dummy cells (denoted by dashed line). The dummy cells (points) are usually not generated as the grid inside the domain (except on multiblock grids). Rather, the cells are only virtual, although there are also geometrical quantities like volume or face vector associated with them.

The purpose of the dummy cells is to simplify the computation of the fluxes, gradients, dissipation, etc. along the boundaries. This is achieved by the possibility to extend the stencil of the spatial discretisation scheme beyond the boundaries. As we can see in Fig. 8.1, the same discretisation scheme can be employed at the boundaries like inside the physical domain. Thus, we can solve the governing equations in the same way in all "physical" grid points. This makes the discretisation scheme much easier to implement. Furthermore, all grid points of a structured grid can be accessed in a single loop, which is of significant advantage particularly on vector computers. The condition is of course that the dummy cells (points) contain appropriate values of the conservative variables as well as of geometrical quantities. Clearly, the number of dummy

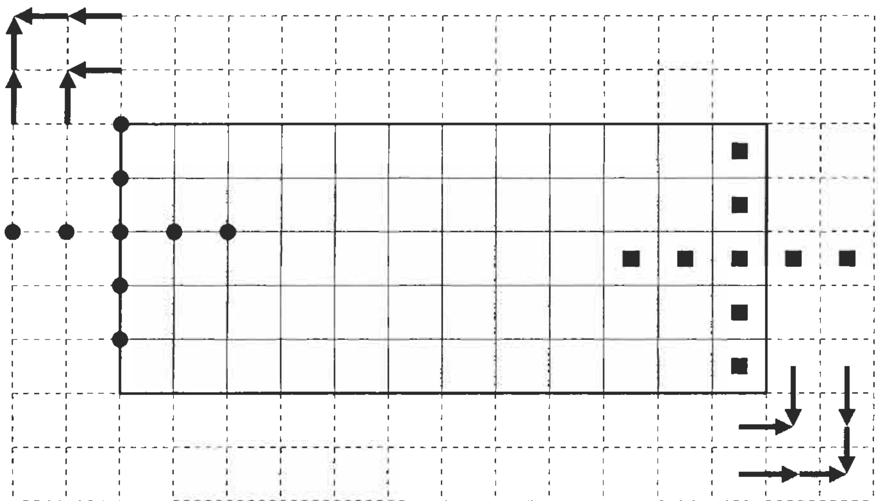


Figure 8.1: Two layers of dummy cells (dashed line) around the computational domain (thick line) in 2D. Filled circles represent the standard stencil of a 2nd-order cell-vertex (dual) scheme, filled rectangles outline the stencil of a 2nd-order cell-centred scheme (see Section 4.3).

cell layers must be such that the part of the stencil outside the physical domain is completely covered. The conservative variables in the dummy cells (points) are obtained from boundary conditions. The geometrical quantities are usually taken from the corresponding control volume at the boundary. In the case of boundaries between multiple grid blocks (Section 3.1), all flow variables and the geometry are transferred from the neighbouring block.

The grey-shaded dummy cells in Fig. 8.1 represent a certain problem, since it is not quite clear how to set their values (if there is no adjacent grid block). The values are not required by the standard cross-type discretisation stencil. However, they may become necessary for the computation of gradients (viscous fluxes - see Section 4.4), or for transfer operators within multigrid (Section 9.4). Usually, an averaging of the values from the adjacent "regular" dummy cells, as indicated in Fig. 8.1 by arrows, is sufficient.

8.2 Solid Wall

8.2.1 Inviscid Flow

In the case of an inviscid flow, the fluid slips over the surface. Since there is no friction force, the velocity vector must be tangent to the surface. This is equivalent to the condition that there is no flow normal to the surface, i.e.,

$$\vec{v} \cdot \vec{n} = 0 \qquad \text{at the surface}\,, \tag{8.1}$$

where $\vec{n}$ denotes the unit normal vector at the surface. Hence, the contravariant velocity V (Eq. (2.22)) is zero at the wall. Consequently, the vector of convective fluxes Eq. (2.21) reduces to the pressure term alone, i.e,

$$(\vec{F}_c)_w = \begin{bmatrix} 0 \\ n_x\, p_w \\ n_y\, p_w \\ n_z\, p_w \\ 0 \end{bmatrix} \tag{8.2}$$

with p_w being the wall pressure.

Structured Cell-Centred Scheme

Within the cell-centred scheme, the pressure is evaluated at the centroid of the cell. However, p_w in Eq. (8.2) is required at the face of the boundary cell. We can obtain the wall pressure most easily by extrapolation from the interior of the domain. Considering Fig. 8.2, we could simply set $p_w = p_2$. Higher accuracy is achieved by using either a two-point

$$p_w = \frac{1}{2}(3p_2 - p_3)\,, \tag{8.3}$$

or a three-point extrapolation formula

$$p_w = \frac{1}{8}(15p_2 - 10p_3 + 3p_4)\,. \tag{8.4}$$

In order to account for grid stretching, distances to the wall could be employed instead of the constant coefficients [1].

The above extrapolation formulae (8.3), (8.4) do not account for the grid and the surface geometry. An alternative approach – the so-called *normal-momentum relation* – was developed by Rizzi [2]. It is based on the fact that the wall represents a streamline in inviscid flow. Differentiation of the zero normal-flow condition in Eq. (8.1) along the surface streamline, and the substitution of the result into the momentum equation yields

$$\rho\vec{v} \cdot (\vec{v} \cdot \vec{\nabla})\,\vec{n} = \vec{n} \cdot \vec{\nabla} p\,. \tag{8.5}$$

Equation (8.5) relates the density, the velocity and the wall geometry to the normal derivative of the pressure. It was demonstrated that the normal-momentum

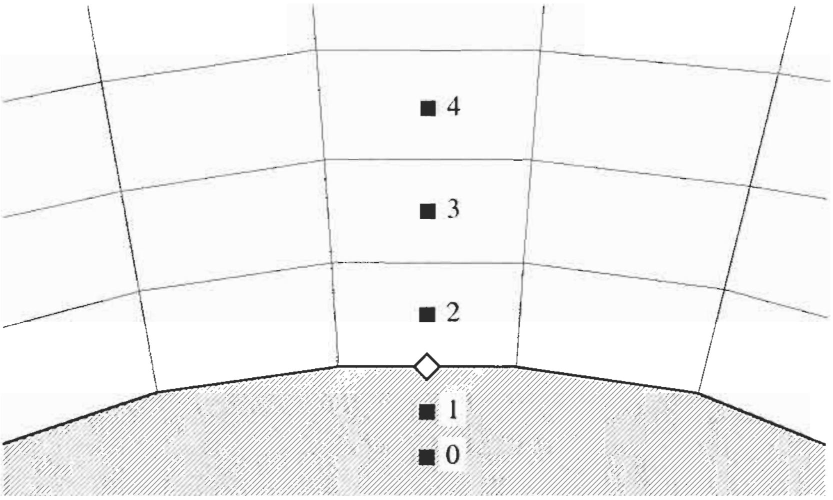


Figure 8.2: Solid wall boundary condition for the cell-centred scheme. Dummy cells are denoted as 0 and 1. Location, where the convective fluxes Eq. (8.2) are evaluated, is marked by a diamond.

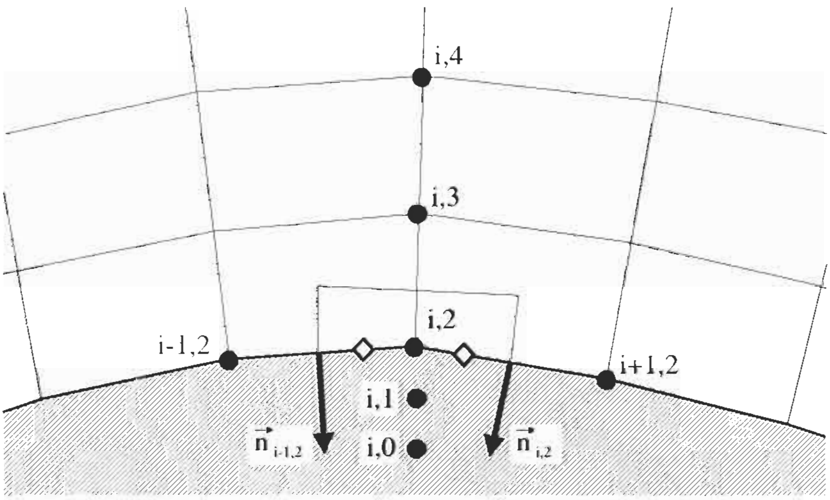


Figure 8.3: Solid wall boundary condition for the structured, cell-vertex dual control-volume scheme. Dummy points are denoted as $(i, 0)$ and $(i, 1)$. Locations, where the convective fluxes Eq. (8.2) are evaluated, are marked by diamonds. Compare also to the sketch in Fig. 4.6.

relation gives very accurate results [1]. However, in the case of complex geometries there can be problems with the numerical solution of the normal-momentum relation. A detailed description of the implementation and accuracy comparisons can be found in Ref. [1].

The values of the conservative variables in the dummy cells can be obtained by linear extrapolation from the interior, i.e.,

$$\begin{aligned}\vec{W}_1 &= 2\vec{W}_2 - \vec{W}_3 \\ \vec{W}_0 &= 3\vec{W}_2 - 2\vec{W}_3\,.\end{aligned} \tag{8.6}$$

The indices in Eq. (8.6) correspond to Fig. 8.2. If the dummy cells are to be utilised within the spatial discretisation scheme (e.g., for the evaluation of the dissipation operator), it is important that the calculation of the convective fluxes is compatible to Eq. (8.2).

Structured Cell-Vertex Scheme

The implementation of the boundary condition Eq. (8.1) is straightforward for the cell-vertex discretisation scheme with overlapping control volumes (Subsection 4.2.2). The convective fluxes at the wall faces are computed according to Eq. (8.2). The wall pressure p_w is obtained by averaging the nodal values as indicated in Eq. (4.26) for a 2-D, or in Eq. (4.27) for a 3-D case. The distribution formula (Eq. (4.30) in 2D) accounts now for only two (four in 3D) cells (compare Fig. 4.4 to Fig. 8.3).

Several different ways can be followed in the case of the cell-vertex scheme with dual control volumes (Subsection 4.2.3). One approach is to apply the condition in Eq. (8.2) separately for each face of the control volume which is on the wall. Thus, according to Fig. 8.3, we can write

$$(\vec{F}_{c,w})_{i,2} = \begin{bmatrix} 0 \\ (n_x)_{i-1,2}\,(p_w)_{i-1/4,2} \\ (n_y)_{i-1,2}\,(p_w)_{i-1/4,2} \\ (n_z)_{i-1,2}\,(p_w)_{i-1/4,2} \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ (n_x)_{i,2}\,(p_w)_{i+1/4,2} \\ (n_y)_{i,2}\,(p_w)_{i+1/4,2} \\ (n_z)_{i,2}\,(p_w)_{i+1/4,2} \\ 0 \end{bmatrix}. \tag{8.7}$$

The pressures $(p_w)_{i-1/4,2}$ and $(p_w)_{i+1/4,2}$ in Eq. (8.7) can be obtained by linear interpolation, e.g.,

$$(p_w)_{i+1/4,2} = \frac{1}{4}\left[3(p_w)_{i,2} + (p_w)_{i+1,2}\right]. \tag{8.8}$$

The corresponding 3-D formula will be presented further below in the subsection on unstructured grids.

Another possible implementation employs the condition in Eq. (8.2) directly in the respective wall node (i.e., node 2 in Fig. 8.3). The wall pressure p_w is simply set equal to p_2. The unit normal vector is computed as the average of the normal vectors of all wall facets which share the node 2. This approach requires a

correction of the velocity vector. After the solution update by the time-stepping scheme, the velocity vector at the wall is projected onto the tangential plane [3], [4], i.e.,

$$(\vec{v}_{i,2})_{corr} = \vec{v}_{i,2} - [\vec{v}_{i,2} \cdot (\vec{n}_{av})_{i,2}] \cdot (\vec{n}_{av})_{i,2} \tag{8.9}$$

with $\vec{n}_{av}$ being the averaged unit normal vector. In this way, the flow will become tangential to the wall.

In order to assign values to the dummy points, it is sufficient to extrapolate the conservative variables (Eq. (2.20)) from the interior field by using relation similar to Eq. (8.6).

Unstructured Cell-Centred Scheme

The wall boundary condition in Eq. (8.1) can be implemented for a cell-centred unstructured scheme in a way similar to that on structured grids. If the boundary cell is a quadrilateral, hexahedron or a prism (with triangular face on the wall), the pressure can be extrapolated to the wall by using Eq. (8.3). The neighbouring cell (number 3 in Fig. 8.2) is known from the face-based data structure described in Subsection 5.2.1. For the case of a triangular or tetrahedral cell, in Ref. [5] and [6] it was suggested to employ one layer of dummy cells. The velocity components in the dummy cells were obtained by reflecting the velocity vectors in the boundary cells at the wall. For example, in the dummy cell 1 in Fig. 8.2, the velocity would become

$$\vec{v}_1 = \vec{v}_2 - 2\,V_2\vec{n}\,, \tag{8.10}$$

where $V_2 = u_2 n_x + v_2 n_y + w_2 n_z$ is the contravariant velocity and $\vec{n} = [n_x, n_y, n_z]^T$ stands for the wall unit-normal vector. The pressure and density in the dummy cells were set equal to the values in the corresponding boundary cell (this implies $p_w = p_2$).

Unstructured Median-Dual Scheme

The boundary condition Eq. (8.1) requires more attention in the case of the median-dual unstructured discretisation scheme. The situation is shown in Fig. 8.4 for the 2-D and in Fig. 8.5 for the 3-D case. The convective fluxes in Eq. (8.2) are computed separately at each face of the control volume which is located on the wall. This is identical to the first approach discussed above for the structured cell-vertex scheme with dual control volumes. For quadrilateral elements (like 1-3-4-5 in Fig. 8.4), the pressure is interpolated correspondingly to Eq. (8.8). In the case of hexahedra, prisms or pyramids, where the face of the control volume is quadrilateral (like face 1-4-5-6 in Fig. 8.5), the interpolation formula reads

$$p_{int} = \frac{1}{16}(9p_1 + 3p_4 + 3p_6 + p_5)\,. \tag{8.11}$$

If the boundary elements are tetrahedra (or triangles in 2-D), the pressure at the wall face should be evaluated like in the finite element method [7]. At the

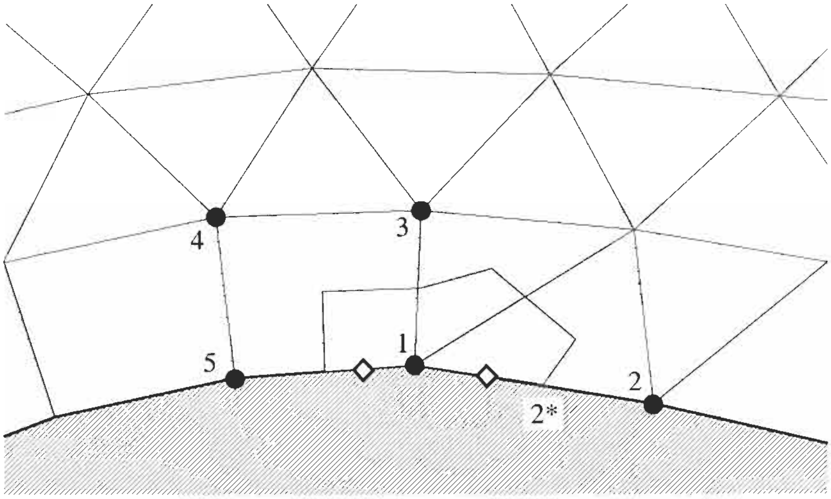


Figure 8.4: Solid wall boundary condition for the 2-D unstructured, dual-control volume mixed-grid scheme. Locations, where the convective fluxes Eq. (8.2) are evaluated, are marked by diamonds.

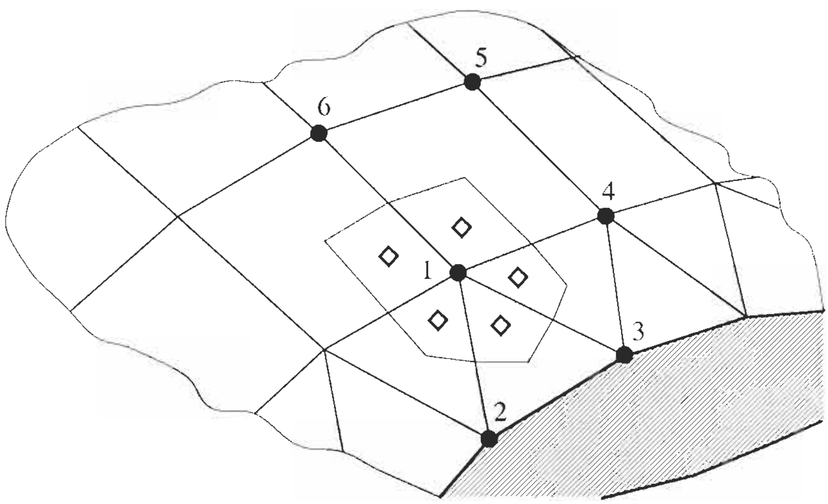


Figure 8.5: Solid wall boundary condition for the 3-D unstructured, mixed-grid scheme. Locations, where the convective fluxes Eq. (8.2) are evaluated, are marked by diamonds.

wall segment 1-2 in Fig. 8.4 for example, the pressure at the face 1-2* would be computed as

$$p_{int} = \frac{1}{6}(5p_1 + p_2)\,. \tag{8.12}$$

In the case of a tetrahedra with, e.g., the wall face 1-2-3 in Fig. 8.5, the pressure is given by

$$p_{int} = \frac{1}{8}(6p_1 + p_2 + p_3)\,. \tag{8.13}$$

8.2.2 Viscous Flow

For a viscous fluid which passes a solid wall, the relative velocity between the surface and the fluid directly at the surface is assumed to be zero. Therefore, we speak of *noslip* boundary condition. In the case of a stationary wall surface, the Cartesian velocity components become

$$u = v = w = 0 \qquad \text{at the surface.} \tag{8.14}$$

There are two basic consequences of the noslip condition. First, we do not need to solve the momentum equations on the wall. This fact is utilised in the cell-vertex scheme. Second, the convective fluxes through the noslip wall are given again by Eq. (8.2), and the terms in Eq. (2.24) simplify to $\vec{\Theta} = k\vec{\nabla}T$. Hence, the wall pressure in the convective fluxes is obtained in the same way as described above for the inviscid flow. However, the dummy cells (points) are treated in a different way.

Cell-Centred Scheme

The implementation of the noslip boundary condition in Eq. (8.14) can be simplified by the utilisation of dummy cells. In the case of an adiabatic wall (no heat flux through the wall), we can set (see Fig. 8.2)

$$\begin{aligned} &\rho_1 = \rho_2\,, \quad E_1 = E_2 \\ &u_1 = -u_2\,, \quad v_1 = -v_2\,, \quad w_1 = -w_2 \end{aligned} \tag{8.15}$$

and likewise for the cells 0 and 3. The approach is applicable to both, structured and unstructured schemes (cf. Ref. [6]).

If the wall temperature is given, the velocity components are still reversed as in Eq. (8.15). The temperature is linearly extrapolated from the interior field by using the specified wall temperature. Since the pressure gradient normal to the wall is zero, the pressure in the boundary element is prescribed also in the dummy cells (i.e., $p_0 = p_1 = p_2$). The density and the total energy in the dummy cells are evaluated from the interpolated values.

Cell-Vertex Scheme

Since the momentum equations need not to be solved, there is no contribution from the convective fluxes (Eq. (8.2)) at the wall. The viscous fluxes in Eq. (2.23) contribute only the temperature gradient normal to the wall to the energy equation. For an adiabatic wall, $\vec{\nabla} T_w \cdot \vec{n}$ is zero. Hence, we do not have to compute any convective or viscous fluxes at the wall. The residuals of the momentum equations should be set to zero, in order to prevent the generation of nonzero velocity components at the wall nodes.

In the case of prescribed wall temperature, we can directly set the total energy at the wall (e.g., node $(i,2)$ in Fig. 8.3) using (perfect gas assumed)

$$(\rho E)_{i,2} = \frac{c_p}{\gamma}\, \rho_{i,2}\, T_w\,, \tag{8.16}$$

where T_w denotes the given wall temperature. The residuals of the momentum and the energy equation have to be zeroed out. The same strategy is applicable also to unstructured schemes.

Another approach, which seems to be more robust for some applications, does not solve the governing equations at the wall at all. Both, the density and the energy are directly specified

$$\rho_{i,2} = \frac{p_{i,3}}{T_w R} \quad \text{and} \quad (\rho E)_{i,2} = \frac{p_{i,3}}{\gamma}\,. \tag{8.17}$$

The relations in Eq. (8.17) assume that there is no pressure gradient normal to the wall (therefore $p_2 = p_3$). Since all conservative variables are prescribed, the residuals of all equations should be set to zero. This technique can be utilised on unstructured grids as well. However, the extrapolation of the pressure requires additional operations on triangular or tetrahedral grids.

If the wall is adiabatic, the values in the dummy points are obtained as follows

$$\begin{aligned} &\rho_{i,1} = \rho_{i,3}\,, \quad E_{i,1} = E_{i,3} \\ &u_{i,1} = -u_{i,3}\,, \quad v_{i,1} = -v_{i,3}\,, \quad w_{i,1} = -w_{i,3}\,. \end{aligned} \tag{8.18}$$

The same applies to the nodes 0 and 4. If the wall temperature is given, the temperature in the dummy points is extrapolated from the interior, i.e.,

$$T_{i,1} = 2T_w - T_{i,3} \quad \text{and} \quad T_{i,0} = 3T_w - 2T_{i,3} \tag{8.19}$$

with the indices according to Fig. 8.3. The velocity components are again reversed as in Eq. (8.18). The density and energy are computed with the interpolated temperature value and with the pressure $p_{i,3}$.

8.3 Farfield

The numerical simulation of external flows past airfoils, wings, cars and other configurations has to be conducted within a bounded domain. For this reason, artificial farfield boundary conditions become necessary. The numerical implementation of the farfield boundary conditions has to fulfil two basic requirements. First, the truncation of the domain should have no notable effects on the flow solution as compared to the infinite domain. Second, any outgoing disturbances must not be reflected back into the flow field [9]. Due to their elliptic nature, sub- and transonic flow problems are particularly sensitive to the farfield boundary conditions. An inadequate implementation can lead to a significant slow down of convergence to the steady state. Furthermore, the accuracy of the solution is likely to be negatively influenced. Various methodologies were developed which are capable of absorbing the outgoing waves at the artificial boundaries [10]-[15]. An review of different non-reflecting boundary conditions can be found in [16].

In the following two subsections, we shall discuss the concept of characteristic variables as it was described by Whitfield and Janus [12]. We shall also present an extension of the farfield boundary conditions for lifting bodies.

8.3.1 Concept of Characteristic Variables

Depending on the sign of the eigenvalues of the convective flux Jacobians (Appendix A.9, Eq. (A.63) or (A.67)), the information is transported out of or into the computational domain along the characteristics. For example, in the case of subsonic inflow there are four incoming characteristics (in 3D) and one outgoing (Λ_5 in Eq. (A.67)). The situation reverses for subsonic outflow. According to the one-dimensional theory of Kreiss [17], the number of conditions to be imposed from outside at the boundary should be equal to the number of **incoming** characteristics. The remaining conditions should be determined from the solution inside the domain.

The approach of Whitfield and Janus [12] is based on the characteristic form of the one-dimensional Euler equations (2.45) normal to the boundary (cf. Appendix A.9). The methodology was found to perform very well on structured and unstructured grids in a variety of flow cases. It can be applied not only to farfield boundaries but also to inviscid solid walls (Subsection 8.2.1).

The two basic flow situations at the farfield boundary are sketched in Fig. 8.6. The flow can either enter or it can leave the domain. Therefore, depending on the local Mach number, four different types of farfield boundary conditions have to be treated:

- supersonic inflow,
- supersonic outflow,
- subsonic inflow, and
- subsonic outflow.

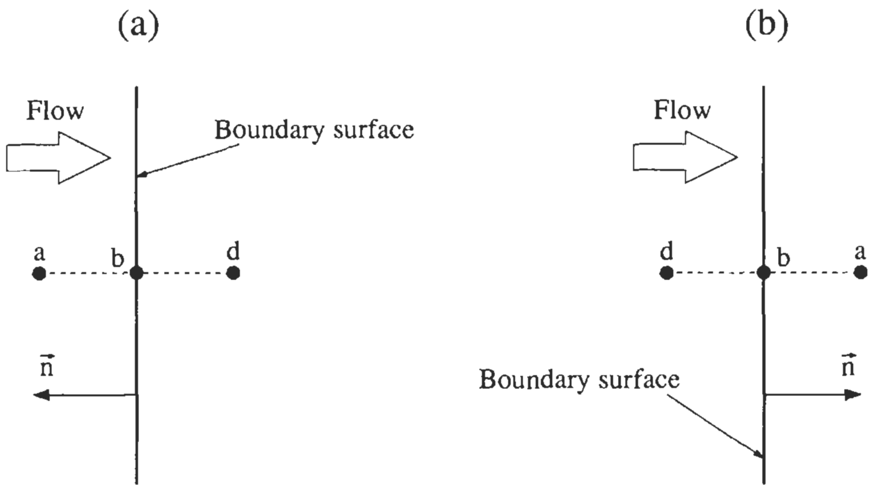


Figure 8.6: Farfield boundary: inflow (a) and outflow (b) situation. Position a is outside, b on the boundary, and position d is inside the physical domain. The unit normal vector $\vec{n} = [n_x, n_y, n_z]^T$ points out of the domain.

Supersonic Inflow

For supersonic inflow, all eigenvalues have the same sign. Since the flow is entering the physical domain, the conservative variables on the boundary (point b in Fig. 8.6) are determined by freestream values only. Thus,

$$\vec{W}_b = \vec{W}_a \,. \tag{8.20}$$

The values $\vec{W}_a$ are specified based on the given Mach number M_∞ and on two flow angles (angle of attack, side-slip angle).

Supersonic Outflow

In this case, all eigenvalues have also the same sign. However, the flow leaves now the physical domain and all conservative variables at the boundary must be determined from the solution inside the domain. This can be accomplished simply by setting

$$\vec{W}_b = \vec{W}_d \,. \tag{8.21}$$

Subsonic Inflow

Here, four characteristics enter and one leaves the physical domain. Therefore, four characteristic variables are prescribed based on the freestream values. One characteristic variable is extrapolated from the interior of the physical domain.

This leads to the following set of boundary conditions [12]

$$
\begin{aligned}
p_b &= \frac{1}{2}\{p_a + p_d - \rho_0 c_0 [n_x(u_a - u_d) + n_y(v_a - v_d) + n_z(w_a - w_d)]\} \\
\rho_b &= \rho_a + (p_b - p_a)/c_0^2 \\
u_b &= u_a - n_x(p_a - p_b)/(\rho_0 c_0) \\
v_b &= v_a - n_y(p_a - p_b)/(\rho_0 c_0) \\
w_b &= w_a - n_z(p_a - p_b)/(\rho_0 c_0)\,,
\end{aligned}
\tag{8.22}
$$

where ρ_0 and c_0 represent reference state. The reference state is normally set equal to the state at the interior point (point d in Fig. 8.6). The values in point a are determined from the freestream state.

Subsonic Outflow

In the case of subsonic outflow, four flow variables (density and the three velocity components) have to be extrapolated from the interior of the physical domain. The remaining fifth variable (pressure) must be specified externally. The primitive variables at the farfield boundary are obtained from [12]

$$
\begin{aligned}
p_b &= p_a \\
\rho_b &= \rho_d + (p_b - p_d)/c_0^2 \\
u_b &= u_d + n_x(p_d - p_b)/(\rho_0 c_0) \\
v_b &= v_d + n_y(p_d - p_b)/(\rho_0 c_0) \\
w_b &= w_d + n_z(p_d - p_b)/(\rho_0 c_0)
\end{aligned}
\tag{8.23}
$$

with p_a being the prescribed static pressure.

Physical properties in the dummy cells can be obtained by linear extrapolation from the states b and d.

8.3.2 Modifications for Lifting Bodies

The above characteristic farfield boundary conditions assume zero circulation, which is not correct for a lifting body in sub- or transonic flow. For this reason, the farfield boundary has to be located very far away from the body. Otherwise, the flow solution will be inaccurate. The distance to the farfield can be significantly shortened (one order of magnitude), if the freestream flow includes the effect of a single vortex (horse-shoe vortex in 3D). The vortex is assumed to be centred at the lifting body. The strength of the vortex is proportional to the lift produced by the body. In the following, we shall present implementations of the vortex correction in 2D and in 3D.

Vortex Correction in 2D

The approach, which we want to describe here, was suggested by Usab and Murman [18]. The components of the corrected freestream velocity are given by the expressions (compressible flow assumed)

$$
\begin{aligned}
u_\infty^* &= u_\infty + \left(\frac{\Gamma\sqrt{1-M_\infty^2}}{2\pi\, d}\right) \frac{1}{1-M_\infty^2\sin^2(\theta-\alpha)}\,\sin\theta \\
v_\infty^* &= v_\infty - \left(\frac{\Gamma\sqrt{1-M_\infty^2}}{2\pi\, d}\right) \frac{1}{1-M_\infty^2\sin^2(\theta-\alpha)}\,\cos\theta
\end{aligned}
\tag{8.24}
$$

with Γ being the circulation, (d,θ) the polar coordinates of the farfield point, α the angle of attack, and M_∞ denoting the freestream Mach number, respectively. The circulation is obtained from

$$
\Gamma = \frac{1}{2}\|\vec{v}_\infty\|_2\, a\, C_L \tag{8.25}
$$

by using the theorem of Kutta-Joukowsky. In Equation (8.25), a represents the chord length of the airfoil and C_L is the lift coefficient evaluated by the integration of the surface pressure. The polar coordinates in Eq. (8.24) are calculated as

$$
\begin{aligned}
d &= \sqrt{(x-x_{ref})^2 + (y-y_{ref})^2} \\
\theta &= \tan\left(\frac{y-y_{ref}}{x-x_{ref}}\right),
\end{aligned}
\tag{8.26}
$$

where x_{ref} and y_{ref} are the coordinates of the reference point (location of the vortex – e.g., at 1/4 chord).

The modified freestream pressure p_∞^* is given by

$$
p_\infty^* = \left[p_\infty^{(\gamma-1)/\gamma} + \left(\frac{\gamma-1}{\gamma}\right)\frac{\rho_\infty\left(\|\vec{v}_\infty\|_2^2 - \|\vec{v}_\infty^*\|_2^2\right)}{2\,p_\infty^{1/\gamma}}\right]^{\gamma/(\gamma-1)} \tag{8.27}
$$

with $\|\vec{v}_\infty^*\|_2^2 = (u_\infty^*)^2 + (v_\infty^*)^2$. The corrected freestream density is obtained from the equation of the state

$$
\rho_\infty^* = \rho_\infty\left(\frac{p_\infty^*}{p_\infty}\right)^{1/\gamma}. \tag{8.28}
$$

The corrected quantities u_∞^*, v_∞^*, p_∞^*, and ρ_∞^* are inserted into Eq. (8.22) or Eq. (8.23) instead of u_a, v_a, p_a, and ρ_a.

The above vortex correction Eqs. (8.24)-(8.28) is strictly valid in subsonic flow only. However, the modification of the freestream conditions proved to be helpful in transonic flow as well. This is demonstrated in Fig. 8.7, where the dependence of the lift coefficient on the distance to the farfield boundary was investigated. The farfield radius was set to 5, 20, 50, and 99 chords. As we

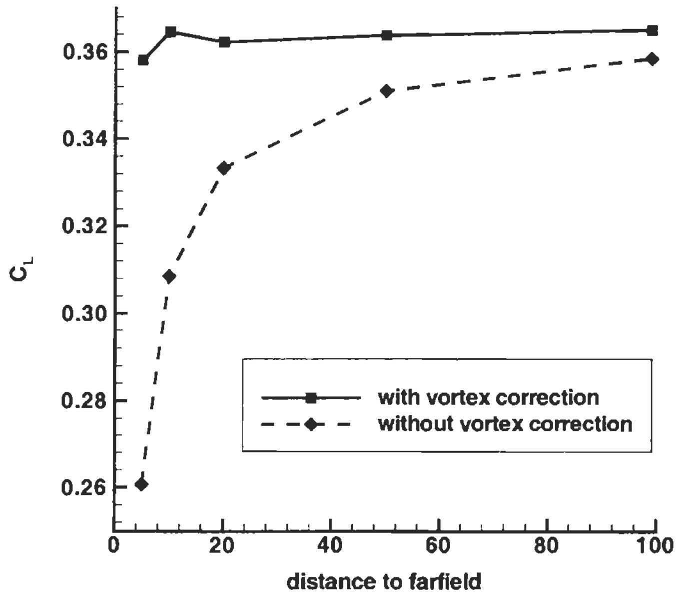


Figure 8.7: Effects of distance to the farfield boundary and of single vortex on the lift coefficient. NACA 0012 airfoil, $M_\infty = 0.8$, $\alpha = 1.25°$.

can see, the simulation without the vortex correction experiences a strong dependence on the farfield distance. On contrary, the calculations with the vortex remain sufficiently accurate up to a distance of about 20 chords. This leads to a significant reduction of the number of grid cells/points. It was demonstrated in Ref. [19] that by using higher-order terms in the vortex correction, the farfield boundary can be placed only about 5 chords away without loss of accuracy.

Vortex Correction in 3D

The effect of a wing on the farfield boundary can be approximated by a horseshoe vortex. In the case of compressible flow, the modified freestream velocity components can be obtained from [20], [21]

$$
\begin{aligned}
u_\infty^* &= u_\infty + \frac{\Gamma\beta^2}{2\pi}\mathcal{A} \\
v_\infty^* &= v_\infty - \frac{\Gamma}{2\pi}\left[\frac{z+l}{(z+l)^2+y^2}\mathcal{B} - \frac{z-l}{(z-l)^2+y^2}\mathcal{C} + \frac{x\beta^2}{x^2+y^2\beta^2}\mathcal{A}\right] \\
w_\infty^* &= w_\infty + \frac{\Gamma}{2\pi}\left[\frac{y}{(z+l)^2+y^2}\mathcal{B} - \frac{y}{(z-l)^2+y^2}\mathcal{C}\right],
\end{aligned}
\tag{8.29}
$$

where Γ denotes the circulation, (x, y, z) the Cartesian coordinates of the farfield point, and l stands for the half span, respectively. Furthermore, in Eq. (8.29) it was assumed that the flow is in the positive x-direction with the wing being oriented along the z-axis. The terms $\mathcal{A}$, $\mathcal{B}$ and $\mathcal{C}$ in Eq. (8.29) read

$$\begin{aligned} \mathcal{A} &= \frac{z+l}{\sqrt{\psi_+}} - \frac{z-l}{\sqrt{\psi_-}} \\ \mathcal{B} &= 1 + \frac{x}{\sqrt{\psi_+}} \\ \mathcal{C} &= 1 + \frac{x}{\sqrt{\psi_-}} \,. \end{aligned} \tag{8.30}$$

The abbreviations are given by

$$\begin{aligned} \psi_+ &= x^2 + \beta^2(z+l)^2 + y^2\beta^2 \\ \psi_- &= x^2 + \beta^2(z-l)^2 + y^2\beta^2 \\ \beta &= \sqrt{1 - M_\infty^2} \end{aligned} \tag{8.31}$$

with M_∞ being the freestream Mach number. The circulation Γ is calculated using Eq. (8.25), where a represents the mean chord. The corrected values of pressure (p_∞^*) and of density (ρ_∞^*) are obtained from the formulae (8.27) and (8.28), respectively. The quantities u_a, v_a, w_a, p_a, and ρ_a in Eq. (8.22) or Eq. (8.23) are replaced by their corrected values u_∞^*, v_∞^*, w_∞^*, p_∞^*, and ρ_∞^*.

The expressions for the corrected velocity components v_∞^* and w_∞^* in Eq. (8.29) becomes infinite at locations, where the vortex lines cross the outflow boundary. These are the points

$$z = +l\,, \quad y = 0\,,$$
$$z = -l\,, \quad y = 0\,,$$

and $x = x_{farf}$. In order to avoid the numerical singularity, in Ref. [21] it was suggested to constrain the values of

$$(z+l)^2 + y^2 \quad \text{and}$$
$$(z-l)^2 + y^2$$

in Eq. (8.29) to the 1/4 wingspan, i.e., $l/2$. This measure reduces the corrections to the velocities v_∞ and w_∞ within the distance $l/2$ around the vortex lines $z = l$ and $z = -l$.

The numerical results presented in [21] indicate a reduced sensitivity of the lift and drag coefficient with respect to the farfield distance, if the vortex correction in Eq. (8.29) is applied. It was found that a distance of $7 \cdot l$ to the farfield boundary is sufficient for accurate results.

8.4 Inlet/Outlet Boundary

Various approaches were devised for the implementation of numerical inlet, and in particular, of outlet (also named open) boundary conditions for the Navier-Stokes equations [22]-[26]. Here, we will concentrate on methodologies, which were developed for turbomachinery applications. Suitable non-reflecting inlet and outlet boundary conditions were described, e.g., in [27]-[30]. Giles [31], and Hirsch and Verhoff [32] suggested non-reflecting boundary conditions for the Euler equations, which are intended for domains with a short distance between the body and the inlet or the outlet plane.

In certain cases, the inlet and outlet boundary are additionally periodic with respect to the velocity as well as the pressure and temperature gradient. This type of flow is encountered, for example, in the simulation of heat exchangers [33]. The implementation of periodic inlet and outlet boundary conditions was presented in [34]-[36] for LES in channels.

Subsonic Inlet

A common procedure consists of the specification of the total pressure, total temperature, and of two flow angles. One characteristic variable has to be interpolated from the interior of the flow domain. One possibility is to employ the outgoing Riemann invariant [30], which is defined as

$$\mathcal{R}^- = \vec{v}_d \cdot \vec{n} - \frac{2\,c_d}{\gamma - 1}\,, \tag{8.32}$$

where the index d denotes the state inside the domain (cf. Fig. 8.6a). The Riemann invariant is used to determine either the absolute velocity or the the speed of sound at the boundary. In practice, it was found that selecting the speed of sound leads to a more stable scheme, particularly for low Mach-number flows. Therefore, we set

$$c_b = \frac{-\mathcal{R}^-(\gamma-1)}{(\gamma-1)\cos^2\theta + 2}\left\{1 + \cos\theta\sqrt{\frac{[(\gamma-1)\cos^2\theta + 2]c_0^2}{(\gamma-1)(\mathcal{R}^-)^2} - \frac{\gamma-1}{2}}\right\} \tag{8.33}$$

with θ being the flow angle relative to the boundary, and c_0 denoting the stagnation speed of sound. Hence,

$$\cos\theta = -\frac{\vec{v}_d \cdot \vec{n}}{\|\vec{v}_d\|_2} \tag{8.34}$$

and

$$c_0^2 = c_d^2 + \frac{\gamma - 1}{2}\|\vec{v}_d\|_2^2\,, \tag{8.35}$$

where $\|\vec{v}_d\|_2$ denotes the total velocity at the interior point d (Fig. 8.6a). The unit normal vector $\vec{n}$ in Eq. (8.34) was assumed to point outwards of the domain.

Quantities like static temperature, pressure, density, or the absolute velocity at the boundary are evaluated as follows

$$
\begin{aligned}
T_b &= T_0 \left(\frac{c_b^2}{c_0^2}\right) \\
p_b &= p_0 \left(\frac{T_b}{T_0}\right)^{\gamma/(\gamma-1)} \\
\rho_b &= \frac{p_b}{RT_b} \\
\|\vec{v}_b\|_2 &= \sqrt{2\, c_p (T_0 - T_b)}\,,
\end{aligned}
\tag{8.36}
$$

where T_0 and p_0 are the given values of total temperature and pressure, R and c_p represent the specific gas constant and the heat coefficient at constant pressure, respectively. The velocity components at the inlet are obtained by decomposing $\|\vec{v}_b\|_2$ according to the two (one in 2D) prescribed flow angles.

Subsonic Outlet

In turbomachinery, the static pressure is usually prescribed at the outlet. The subsonic outlet boundary can be treated in a way quite similar to the outflow condition in Eq. (8.23). Only the ambient pressure p_a is replaced here by the given static exit pressure.

Flow variables in the dummy cells can be obtained by linearly extrapolating the states at the boundary and at the interior point d.

8.5 Symmetry Plane

If the flow is to be symmetrical with respect to a line or a plane, the first condition which must be met is that there is no flux across the boundary. This is equivalent to the requirement that the velocity normal to the symmetry boundary is zero. Furthermore, the following gradients have to vanish:

- gradient normal to boundary of a scalar quantity,
- gradient normal to boundary of a tangential velocity,
- gradient along the boundary of the normal velocity (since $\vec{v} \cdot \vec{n} = 0$).

We can write these conditions as

$$\begin{aligned} \vec{n} \cdot \vec{\nabla} U &= 0 \\ \vec{n} \cdot \vec{\nabla}(\vec{v} \cdot \vec{t}) &= 0 \\ \vec{t} \cdot \vec{\nabla}(\vec{v} \cdot \vec{n}) &= 0\,, \end{aligned} \tag{8.37}$$

where U stands for a scalar variable and $\vec{t}$ denotes a vector tangential to the symmetry boundary.

Cell-Centred Scheme

The implementation of the symmetry boundary condition can be largely simplified by employing dummy cells. The flow variables in the dummy cells are obtained using the concept of *reflected cells*. This means that scalar quantities like density or pressure in the dummy cells are set equal to the values in the opposite interior cells, i.e.,

$$U_1 = U_2 \quad \text{and} \quad U_0 = U_3\,. \tag{8.38}$$

The notation corresponds to that in Fig. 8.2. The velocity components are reflected with respect to the boundary as indicated in Eq. (8.10). The normal gradient of the normal velocity in the dummy cell equals to that in the opposite interior cell, but it has reversed sign.

Cell-Vertex Scheme (Dual Control Volume)

Two different approaches can be followed. One possibility is to construct the missing half of the control volume by mirroring the grid on the boundary. The fluxes and the gradients are then evaluated like in the interior using reflected flow variables (see above). The second methodology computes the fluxes for the half control volume (but not across the boundary). The components normal to the symmetry plane of the residual are then zeroed out. It is also necessary to correct normal vectors of those faces of the control volume, which touch the boundary (like at point 2* in Fig. 8.4). The modification consists of removing all components of the face vector, which are normal to the symmetry plane. The gradients have also to be corrected according to Eq. (8.37).

8.6 Coordinate Cut

This type of boundary condition is encountered only in the case of structured grids. The coordinate cut represents an artificial, not a physical, boundary. It is a line (plane in 3D) composed of grid points with different computational coordinate(s) but the same physical location. This means that the grid is folded such that it touches itself. As we shall see in Subsection 11.1.1, the coordinate cut appears for the so-called C- (Fig. 11.5) or O-grid topology (Fig. 11.9). The flow variables and their gradients have to stay continuous across the cut.

The best way to implement the cut boundary condition is to employ dummy cells (points). The situation is sketched in Fig. 8.8. As we can see, the dummy layers here are not virtual, but they coincide with the grid on the opposite side of the cut. Hence, the values of physical quantities in the dummy cells (cell-centred scheme), or in the dummy points (cell-vertex scheme), are obtained directly from the opposite cells (points). In the case of the cell-centred scheme, the fluxes across the faces of the boundary cell (shaded in Fig. 8.8a) are evaluated exactly like in the interior field.

The cut boundary can be treated in two different ways for the cell-vertex scheme. One possibility is to generate a complete control volume at the cut (the second part is denoted by a dashed line in Fig. 8.8b). Using the dummy points, the fluxes can be calculated in the same way as inside the domain. If the implementation is done correctly, the flow quantities at the points 2 (upper grid part) and 5 (lower part) will be equal. The second approach is to integrate the fluxes separately for each half of the control volume. The residuals at the points 2 and 5 in Fig. 8.8b are then added. It is important that the partial control volumes at the points 2 and 5 are summed as well.

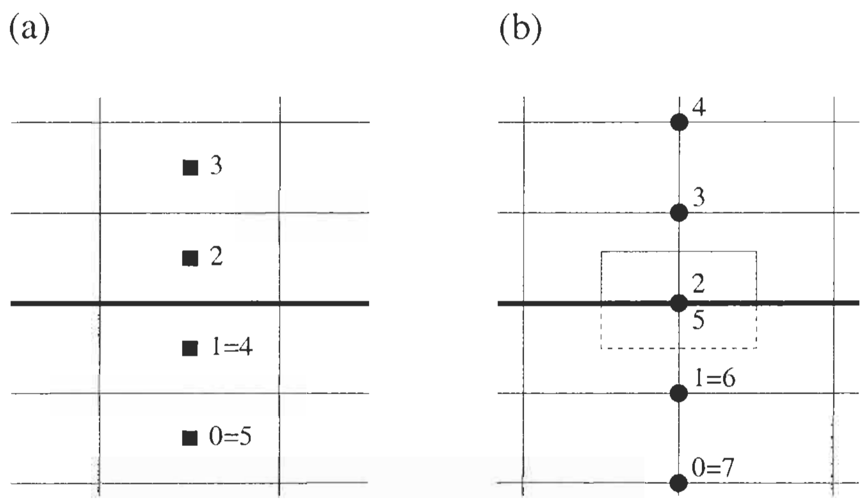


Figure 8.8: Coordinate cut (thick line): cell-centred scheme (a), dual control-volume scheme (b). Dummy cells (points) are numbered as 0 and 1.

8.7 Periodic Boundaries

There are certain practical applications where the flow field is periodic with respect to one or multiple coordinate directions. In such a case, it is sufficient to simulate the flow only within one of the repeating regions. The correct interaction with the remaining physical domain is enforced via periodic boundary conditions.

We can distinguish between two basic types of periodic boundaries. The first one covers *translational* periodicity. This means that one periodic boundary can be transformed into the other boundary by pure coordinate translation. The second type represents periodic boundaries, which were generated by coordinate rotation. Thus, we speak of *rotational* periodicity.

In the following, we shall describe the implementation of the periodic boundary conditions for the cell-centred and the cell-vertex scheme. We shall also consider the case of rotational periodicity. Further details of the treatment of periodic boundaries can be found in Refs. [37], [38].

Cell-Centred Scheme

The utilisation of the dummy-cells concept enables a simple implementation of the periodic boundary condition. Let us consider the example from turbomachinery in Fig. 8.9. The configuration is periodic in the vertical direction. The shaded cells 1 and 2 are located on the lower and the upper periodic boundary, respectively. Due to the periodicity condition, the first dummy-cell layer corresponds to the boundary cells at the opposite periodic boundary. The second dummy-cell layer communicates with the second layer of the physical cells and so on. Hence, all scalar quantities (density, pressure, etc.) in the dummy cells are obtained directly from the corresponding physical cells, i.e,

$$U_{1'} = U_1 \quad \text{and} \quad U_{2'} = U_2 \,. \tag{8.39}$$

The same relations hold also for the vector quantities (velocity, gradients) in the case of translational periodicity. Rotational-periodic boundaries require a correction of the vector variables. This will be discussed further below.

Cell-Vertex Scheme (Dual Control Volume)

This situation is sketched in Fig. 8.10. One approach for the treatment of periodic boundaries consists of the integration of the fluxes around the faces of the shaded control volumes. The residuals at the points 1 and 2 in Fig. 8.10 are then summed in order to obtain the complete net flux. Thus,

$$\vec{R}_{1,\,sum} = \vec{R}_1 + \vec{R}_{2'} \quad \text{and} \quad \vec{R}_{2,\,sum} = \vec{R}_2 + \vec{R}_{1'} \,. \tag{8.40}$$

The partial control volumes at the points 1 and 2 (shaded in Fig. 8.10) have to be added up as well. In the case of translational periodicity, the residuals from the opposite boundary remain unchanged, i.e., $\vec{R}_{1'} = \vec{R}_1$ and $\vec{R}_{2'} = \vec{R}_2$. This results in $\vec{R}_{1,\,sum} = \vec{R}_{2,\,sum}$. Rotationally periodic boundaries require a transformation of the momentum equations before Eq. (8.40) can be applied.

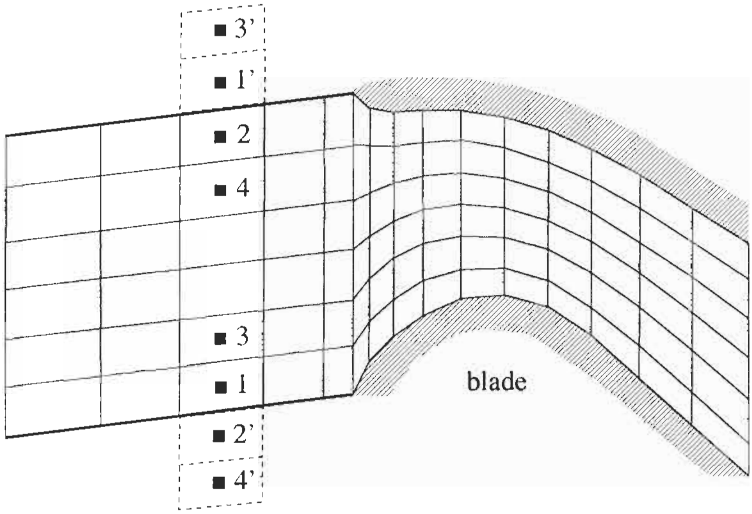


Figure 8.9: Periodic boundaries (thick lines) in the case of 2-D un-/structured, cell-centred scheme. Dummy cells (dashed line) are denoted by the (primed) numbers of the corresponding physical cells.

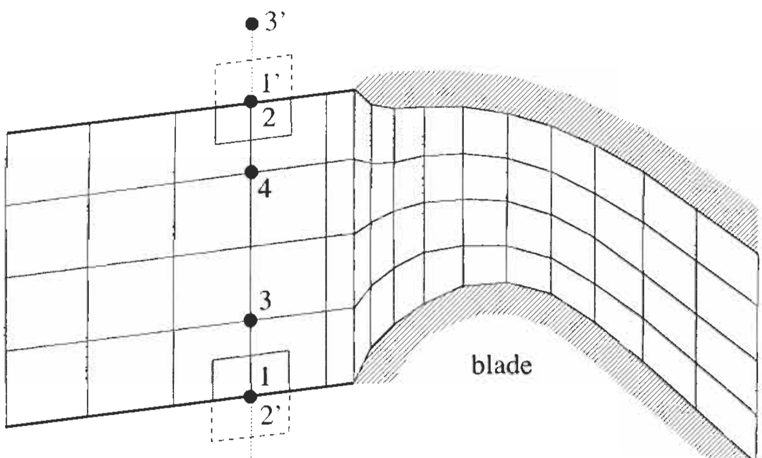


Figure 8.10: Periodic boundaries (thick lines) in the case of 2-D un-/structured, cell-vertex scheme with dual control volumes. The "dummy" parts of the control volumes (dashed line) are denoted by the (primed) numbers of the corresponding control volumes at the opposite boundary. The same holds also for the dummy points 3' and 4'.

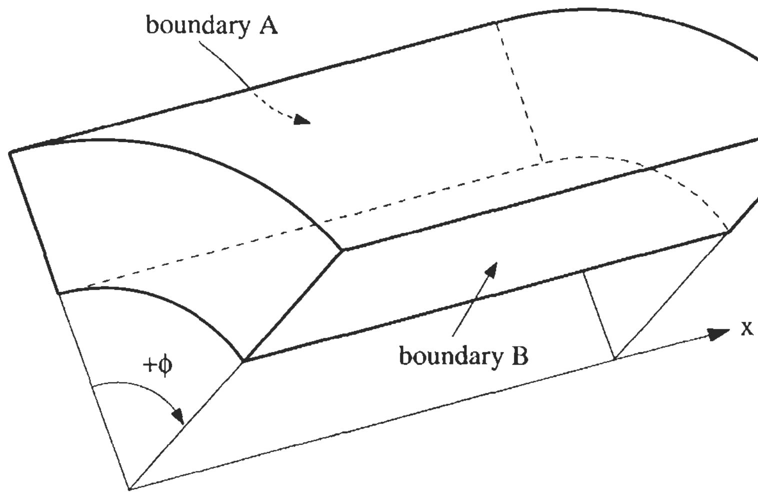


Figure 8.11: Rotationally periodic boundaries (A and B). The rotational axis is assumed to coincide with the x-axis.

Rotational Periodicity

The rotational periodicity condition is based on a rotation of the coordinate system. Therefore, all vector quantities like velocity or gradients of scalars have to be transformed accordingly. Scalar quantities like pressure or density, which are invariant with respect to coordinate rotation, remain unchanged. If we assume the rotational axis is parallel to the x-axis (see Fig. 8.11), the rotation matrix becomes

$$\bar{\mathcal{R}} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix}, \tag{8.41}$$

where the angle ϕ between the periodic boundaries A and B is positive in the clockwise direction. Hence, for example, the velocity vector transformed from boundary A to B (cells 1', 2' in Fig. 8.9 and points 1'–4' in Fig. 8.10) reads

$$\vec{v}_B = \bar{\mathcal{R}}\,\vec{v}_A\,. \tag{8.42}$$

It is easy to show that the x-component of $\vec{v}_A$ (i.e., u_A) is not changed by the rotation. Thus, $u_B = u_A$. The gradients of all flow quantities are transformed in similar way.

As stated above, in the case of the cell-vertex scheme the residuals of the momentum equations must be corrected before the summation in Eq. (8.40) can take place. The application of the rotation matrix Eq. (8.41) leads to

$$\vec{R}^{u,v,w}_{B,\,sum} = \vec{R}^{u,v,w}_{B} + \bar{\mathcal{R}}\,\vec{R}^{u,v,w}_{A}\,. \tag{8.43}$$

The superscript ^{u,v,w} in Eq. (8.43) denotes the three momentum equations.

8.8 Interface Between Grid Blocks

During the discussion of the spatial discretisation with structured grids in Section 3.1, it became evident that it is usually not possible to generate a single grid inside a geometrically complex domain (see Fig. 3.4). We mentioned two possible methodologies how to solve the problem. The first one was the multiblock approach and the second one was the Chimera technique. In the following, we shall describe the basic implementation issues of the multiblock approach. For more throughout discussion, the reader is referred to Refs. [39]-[45]. A very helpful introduction to the multiblock methodology was presented in [46]. Details of the Chimera technique, which is not treated here, can be found in Refs. [47]-[51].

Within the multiblock technique, the physical domain is split into a certain number of virtual parts. Consequently, the computational domain becomes also divided into the same number of blocks. In a general case, the physical solution in a particular block will depend on the flow in one or multiple neighbouring blocks. Therefore, we have to provide a data structure which allows for an efficient exchange of information between the blocks. The structure is also required for communication, if different processors are used to solve the governing equations in the blocks.

The first part of the data structure consists of the numbering of the block boundaries. One particular numbering scheme is displayed in Fig. 8.12. The numbering strategy in Fig. 8.12 can be summarised as follows:

$$\begin{aligned}
&\text{boundary } 1: && i = IBEG \\
&\text{boundary } 2: && i = IEND \\
&\text{boundary } 3: && j = JBEG \\
&\text{boundary } 4: && j = JEND \\
&\text{boundary } 5: && k = KBEG \\
&\text{boundary } 6: && k = KEND\,.
\end{aligned}$$

It is important that all blocks employ the same numbering scheme. The indices i, j, k of the grid points in the computational space are defined in the ranges

$$\begin{aligned}
IBEG \le i \le IEND \\
JBEG \le j \le JEND \\
KBEG \le k \le KEND\,.
\end{aligned}$$

The cell indices I, J, K, which are required by the cell-centred scheme are defined in a similar way. Since the multiblock approach is usually implemented using dummy cells/points, the physical cells/points will have a certain offset from the start or the end of each range (see Fig. 8.1).

The boundary of each block is divided into a number of non-overlapping patches. This allows the specification of different boundary conditions on the same block boundary. The situation is depicted in Fig. 8.13. For a unique identification of each patch it is necessary to store the number of the corresponding block and the number of the block boundary. Furthermore, the origin, the height

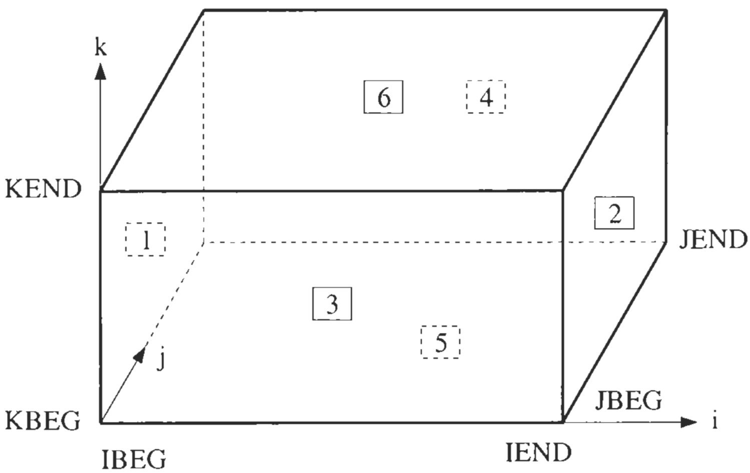


Figure 8.12: Numbering of the sides of the computational space and of the block boundaries.

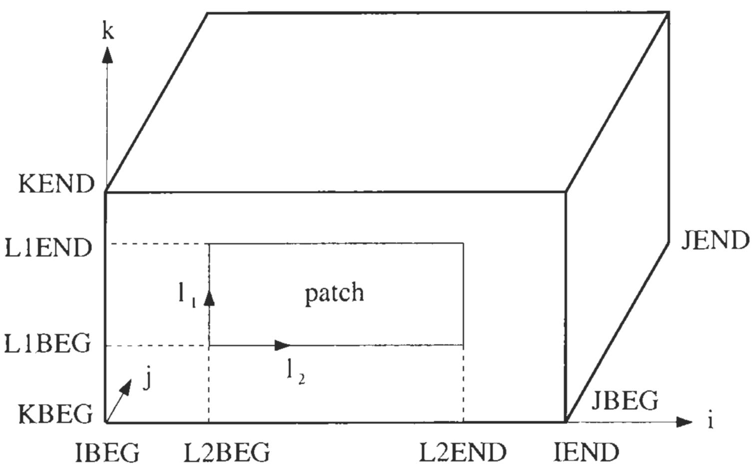


Figure 8.13: Coordinates of a boundary patch in computational space. The patch has its own local coordinate system l_1, l_2.

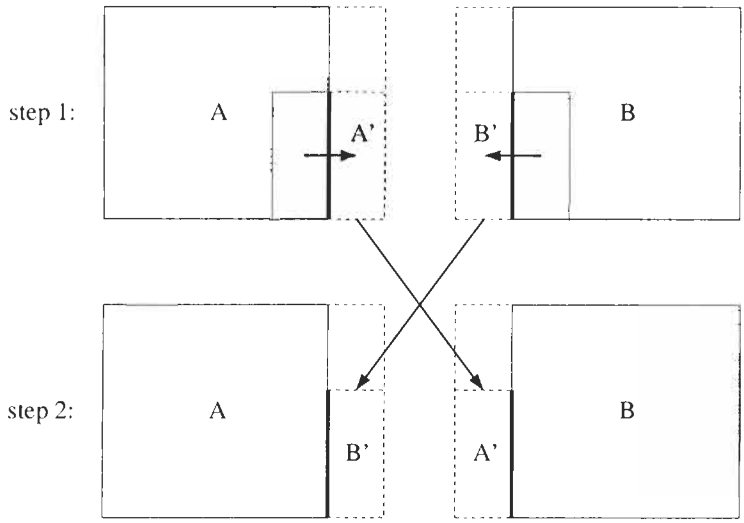


Figure 8.14: Exchange of flow variables (in shaded regions A', B') between two blocks A and B. Dummy layers are denoted by a dashed line.

and the width of the patch must be stored. For this purpose, the coordinates *L1BEG*, *L1END*, *L2BEG* and *L2END* are used in Fig. 8.13. It is suggested to orient the coordinate system of the patch according to the *cyclic directions*. This means, that if we consider the i-coordinate, j and k will be the first and the second cyclic direction. In the case of the j-coordinate, the cyclic directions will become k and i, respectively. Therefore, since the patch in Fig. 8.13 is on the $j = JBEG$ boundary, the l_1-coordinate is oriented in the k-direction and l_2 in the i-direction. The application of the cyclic directions allows for a unique definition of the patch orientation.

The remaining part of the data structure makes sure that data can be exchanged between those patches, which represent interfaces between the blocks (we assume that the blocks communicate only across their faces). For this purpose, it is required to extend the above patch data structure by the numbers of the adjacent block and patch.

The exchange of flow quantities between two blocks is sketched in Fig. 8.14. The procedure consists of two steps. In the first step, variables from the part of the domain, which is overlapped by the dummy layers of the adjacent patch are written to the own dummy cells/points or to a temporary storage (A' and B' in Fig. 8.14). This is done for all blocks. In the second step, the data in A' and B' is exchanged between both blocks. This means that A' is written to the dummy layers of block B and B' to the dummy layers of block A. If the two patches have a different orientation, the data must be transformed accordingly. In cases where the grid lines do not match at the block interface, further operations are required as described, e.g., in [52], [53].

8.9 Flow Gradients at Boundaries of Unstructured Grids

We already stated in Subsection 5.3.4 that the evaluation of the flow gradients requires some care in the case of the median-dual scheme. If the gradients are calculated on triangular or tetrahedral grids using the Green-Gauss approach in Eq. (5.46), the contributions from the boundaries of the domain (except at symmetry or periodic boundaries) must be evaluated similar to Eq. (8.12) or Eq. (8.13) instead of the arithmetic average. Otherwise, the gradient will not be accurate. Considering the notation in Fig. 8.4, the contribution to the boundary node 1 reads

$$\frac{1}{6}(5U_1 + U_2)\,\vec{n}_{12}\,\frac{\Delta S_{12}}{2}\,,$$

where ΔS_{12} is the length of the boundary face between node 1 and 2 (therefore halved). Corresponding to Eq. (8.13), the contribution of the triangular face 1-2-3 to node 1 in Fig. 8.5 becomes

$$\frac{1}{8}(6U_1 + U_2 + U_3)\,\vec{n}_{123}\,\frac{\Delta S_{123}}{3}$$

with $\Delta S_{123}/3$ being the grey area in the triangle 1-2-3. On mixed grids, it is more appropriate to employ the least squares approach with virtual edges [8] (see Fig. 5.15).

The cell-centred scheme requires no special provisions at symmetry or periodic boundaries. The implementation is identical to that discussed for the fluxes in Section 8.5 or 8.7. This holds also for the median-dual scheme, if the gradients are evaluated using the least-squares approach. The only additional work required is to set certain gradients to zero as described previously in Section 8.5 (cf. Eq. (8.37)).

If the Green-Gauss approach is employed within the median-dual scheme (i.e., if Eq. (5.46) is applied), it is necessary to correct normal vectors of those faces of the control volume, which touch the boundary (like at point 2^* in Fig. 8.4). This is done by setting all components of the face vector to zero, which are normal to the symmetry plane. Finally, the gradients are corrected as discussed in Section 8.5. At periodic boundaries, the gradients and the volumes from both sides of the boundary have to be summed up as presented in Section 8.7 for the fluxes (Eq. (8.40)). In the case of rotational periodicity, the gradients needs to be transformed by applying the rotation matrix in Eq. (8.41).

Bibliography

[1] Kroll, N.; Jain, R.K.: *Solution of Two-Dimensional Euler Equations - Experience with a Finite Volume Code.* DFVLR-FB 87-41, 1987.

[2] Rizzi, A.: *Numerical Implementation of Solid-Body Boundary Conditions for the Euler Equations.* ZAMM, 58 (1978), pp. 301-304.

[3] Hall, M.G.: *Cell Vertex Multigrid Scheme for Solution of the Euler Equations.* Proc. Conf. on Numerical Methods for Fluid Dynamics, Reading, UK, 1985.

[4] Koeck, C.: *Computation of Three-Dimensional Flow Using the Euler Equations and a Multiple-Grid Scheme.* Int. Journal for Numerical Methods in Fluids, 5 (1985), pp. 483-500.

[5] Frink, N.T.; Parikh, P.; Pirzadeh, S.: *A Fast Upwind Solver for the Euler Equations on Three-Dimensional Unstructured Meshes.* AIAA Paper 91-0102, 1991.

[6] Frink, N.T.: *Recent Progress Toward a Three-Dimensional Navier-Stokes Solver.* AIAA Paper 94-0061, 1994.

[7] Luo, H.; Baum, J.D.; Löhner, R.: *An Improved Finite Volume Scheme for Compressible Flows on Unstructured Grids.* AIAA Paper 95-0348, 1995.

[8] Haselbacher, A.; Blazek, J.: *On the Accurate and Efficient Discretisation of the Navier-Stokes Equations on Mixed Grids.* AIAA Paper 99-3363, 1999; also AIAA Journal, 38 (2000), pp. 2094-2102.

[9] Mazaheri, K.; Roe, P.L.: *Numerical Wave Propagation and Steady-State Solutions: Soft Wall and Outer Boundary Conditions.* AIAA Journal, 35 (1997), pp. 965-975.

[10] Engquist, B.; Majda, A.: *Absorbing Boundary Conditions for Numerical Simulation of Waves.* Mathematics of Computations, 31 (1977), pp. 629-651.

[11] Bayliss, A.; Turkel, E.: *Far Field Boundary Conditions for Compressible Flow.* J. Computational Physics, 48 (1982), pp. 182-199.

[12] Whitfield, D.L.; Janus, J.M.: *Three-Dimensional Unsteady Euler Equations Solution Using Flux Vector Splitting.* AIAA Paper 84-1552, 1984.

[13] Gustafsson, B.: *Far Field Boundary Conditions for Time-Dependent Hyperbolic Systems.* Center for Large Scale Sci. Comput., CLaSSiC-87-16, Stanford University, 1987.

[14] Thompson, K.W.: *Time Dependent Boundary Conditions for Hyperbolic Systems.* J. Computational Physics, 68 (1987), pp. 1-24.

[15] Hayder, M.E.; Hu, F.Q.; Hussaini, M.Y.: *Towards Perfectly Absorbing Boundary Conditions for Euler Equations.* ICASE Report No. 97-25, 1997.

[16] Givoli, D.: *Non-Reflecting Boundary Conditions.* J. Computational Physics, 94 (1991), pp. 1-29.

[17] Kreiss, H.O.: *Initial Boundary Value Problems for Hyperbolic Systems.* Comm. Pure Appl. Math., 23 (1970), pp. 277-298.

[18] Usab, W.J.; Murman, E.M.: *Embedded Mesh Solution of the Euler Equation Using a Multiple-Grid Method.* AIAA Paper 83-1946, 1983.

[19] Giles, M.B.; Drela, M.; Thompkins, W.T.: *Newton Solution of Direct and Inverse Transonic Euler Equations.* AIAA Paper 85-1530, 1985.

[20] Klunker, E.B.; Harder, K.C.: *Notes on Linearized Subsonic Wing Theory.* Unpublished.

[21] Radespiel, R.: *A Cell-Vertex Multigrid Method for the Navier-Stokes Equations.* NASA TM-101557, 1989.

[22] Papanastasiou, T.C.; Malamataris, N.; Ellwood, K.: *A New Outflow Boundary Condition.* Int. J. Numerical Methods in Fluids, 14 (1992), pp. 587-608.

[23] Sani, R.L.; Gresho, P.M.: *Résumé and Remarks on the Open Boundary Condition Minisymposium.* Int. J. Numerical Methods in Fluids, 18 (1994), pp. 983-1008.

[24] Baum, M.; Poinsot, T.; Thevenin, D.: *Accurate Boundary Conditions for Multicomponent Reactive Flows.* J. Computational Physics, 116 (1994), pp. 247-261.

[25] Griffiths, D.F.: *The 'No Boundary Condition' Outflow Boundary Condition.* Int. J. Numerical Methods in Fluids, 24 (1997), pp. 393-411.

[26] Renardy, M.: *Imposing 'No'Boundary Condition at Outflow: Why Does it Work?.* Int. J. Numerical Methods in Fluids, 24 (1997), pp. 413-417.

[27] Rudy, D.H.; Strikwerda, J.C.: *A Nonreflective Outflow Boundary Condition for Subsonic Navier-Stokes Calculations.* J. Computational Physics, 36 (1980), pp. 55-70.

[28] Yokota, J.W.; Caughey, D.A.: *An L-U Implicit Multigrid Algorithm for the Three-Dimensional Euler Equations.* AIAA Paper 87-0453, 1987.

[29] Yokota, J.W.: *Diagonally Inverted Lower-Upper Factored Implicit Multigrid Scheme for the Three-Dimensional Navier-Stokes Equations.* AIAA Journal, 28 (1989), pp. 1642-1649.

[30] Holmes, D.G.: *Inviscid 2D Solutions on Unstructured, Adaptive Grids.* Numerical Methods for Flows in Turbomachinery, VKI-LS 1989-06, 1989.

[31] Giles, M.B.: *Non-Reflecting Boundary Conditions for Euler Equation Calculations.* AIAA Paper 89-1942, 1989.

[32] Hirsch, Ch.; Verhoff, A.: *Far Field Numerical Boundary Conditions for Internal and Cascade Flow Computations.* AIAA Paper 89-1943, 1989.

[33] Patankar, S.V.; Liu, C.H.; Sparrow, E.M.: *Fully Developed Flow and Heat Transfer in Ducts Having Streamwise-Periodic Variations of Cross-Sectional Area.* ASME J. Heat Transfer, 99 (1977), pp. 180-186.

[34] Deschamps, V.: *Simulations numériques de la turbulence inhomogène incompressible dans un écoulement de canal plan.* ONERA, note technique 1988-5, 1988.

[35] Mossi, M.: *Simulation of Benchmark and Industrial Unsteady Compressible Turbulent Fluid Flows.* PhD Thesis, No. 1958, École Polytechnique Fédérale de Lausanne, Département de Génie Mécanique, Switzerland, 1999, pp. 136-137.

[36] Lenormand, E.; Sagaut, P.; Phuoc, L.T.; Comte, P.: *Subgrid-Scale Models for Large-Eddy Simulations of Compressible Wall Bounded Flows.* AIAA Journal, 38 (2000), pp. 1340-1350.

[37] Chung, H.-T.; Baek, J.-H.: *Influence of Trailing-Edge Grid Structure on Navier-Stokes Computation of Turbomachinery Cascade Flow.* Int. J. Numerical Methods in Fluids, 15 (1992), pp. 883-894.

[38] Segal, G.; Vuik, K.; Kassels, K.: *On the Implementation of Symmetric and Antisymmetric Periodic Boundary Conditions for Incompressible Flow.* Int. J. Numerical Methods in Fluids, 18 (1994), pp. 1153-1165.

[39] Rossow, C.-C.: *Efficient Computation of Inviscid Flow Fields Around Complex Configurations Using a Multiblock Multigrid Method.* Communications in Applied Numerical Methods, 8 (1992), pp. 735-747.

[40] Jacquotte, O.P.: *Generation, Optimization and Adaptation of Multiblock Grids Around Complex Configurations in Computational Fluid Dynamics.* Int. J. Numerical Methods in Engineering, 34 (1992), pp. 443-454.

[41] Szmelter, J.; Marchant, M.J.; Evans, A.; Weatherill, N.P.: *Two-Dimensional Navier-Stokes Equations with Adaptivity on Structured Meshes.* Computer Methods in Applied Mechanics and Engineering, 101 (1992), pp. 355-368.

[42] Marchant, M.J.; Weatherill, N.P.: *The Construction of Nearly Orthogonal Multiblock Grids for Compressible Flow Simulation.* Communications in Numerical Methods in Engineering, 9 (1993), pp. 567-578.

[43] Jenssen, C.B.: *Implicit Multiblock Euler and Navier-Stokes Calculations.* AIAA Journal, 32 (1994), pp. 1808-1814.

[44] De-Nicola, C.; Pinto, G.; Tognaccini, R.: *On the Numerical Stability of Block Structured Algorithms with Applications to 1-D Advection-Diffusion Problems.* Computers & Fluids, 24 (1995), pp. 41-54.

[45] Enander, R.; Sterner, E.: *Analysis of Internal Boundary Conditions and Communication Strategies for Multigrid Multiblock Methods.* Report No. 191, Dept. Sci. Computing, Uppsala University, Sweden, Jan. 1997.

[46] Rizzi, A.; Eliasson, P.; Lindblad, I.; Hirsch, C.; Lacor, C.; Haeuser, J.: *The Engineering of Multiblock/Multigrid Software for Navier-Stokes Flows on Structured Meshes.* Computers & Fluids, 22 (1993), pp. 341-367.

[47] Benek, J.A.; Buning, P.G.; Steger, J.L.: *A 3-D Chimera Grid Embedding Technique.* AIAA Paper 85-1523, 1985.

[48] Buning, P.G.; Chu, I.T.; Obayashi, S.; Rizk, Y.M.; Steger, J.L.: *Numerical Simulation of the Integrated Space Shuttle Vehicle in Ascent.* AIAA Paper 88-4359-CP, 1988.

[49] Chesshire, G.; Henshaw, W.D.: *Composite Overlapping Meshes for the Solution of Partial Differential Equations.* J. Computational Physics, 90 (1990), pp. 1-64.

[50] Pearce, D.G.; et al.: *Development of a Large Scale Chimera Grid System for the Space Shuttle Launch Vehicle.* AIAA Paper 93-0533, 1993.

[51] Kao, H.-J.; Liou, M.-S.; Chow, C.-Y.: *Grid Adaptation using Chimera Composite Overlapping Meshes.* AIAA Journal, 32 (1994), pp. 942-949.

[52] Rai, M.M.: *A Conservative Treatment of Zonal Boundaries for Euler Equation Calculations.* J. Computational Physics, 62 (1986), pp. 472-503.

[53] Kassies, A.; Tognaccini, R.: *Boundary Conditions for Euler Equations at Internal Block Faces of Multi-Block Domains Using Local Grid Refinement.* AIAA Paper 90-1590, 1990.

Chapter 9

Acceleration Techniques

Various methodologies were developed in order to accelerate the solution of the governing equations (2.19) for stationary problems. The acceleration techniques are applicable also to the inner iteration of the dual-time stepping scheme (Section 6.3). The following methods will be discussed in this Chapter:

1. local time-stepping,
2. enthalpy damping,
3. residual smoothing,
4. multigrid,
5. preconditioning.

The local time-stepping, enthalpy damping, and preconditioning are based on a modification of the system of the ordinary differential equations (6.1), while the two remaining techniques are improvements of the solution process. With the exception of the residual smoothing, all methods can be applied to both the explicit (Section 6.1) and the implicit (Section 6.2) time-stepping schemes. Residual smoothing was developed especially for the explicit multistage schemes (Subsections 6.1.1, 6.1.2). An overview of several acceleration techniques can be found in Refs. [1] and [2].

9.1 Local Time-Stepping

In this case, the discretised governing equations (6.1) are integrated using the largest possible time step for each control volume. The local time step Δt_I is calculated according to one of the formulae (6.14), (6.18), (6.20), or (6.22). As a result, the convergence to the steady state is strongly accelerated, but the transient solution is no longer temporally accurate.

9.2 Enthalpy Damping

In certain cases, the total enthalpy H (Eq. (2.12)) is constant in the whole flow field. This situation occurs, for example, for external flows governed by the Euler equations in absence of heat sources and external forces. We can take advantage of this fact in order to reduce the computational effort.

The first possibility would be to prescribe the value of the total enthalpy in the flow domain. In consequence, the energy equation can be omitted from the Euler equations (2.45). This saves memory and CPU time.

A different methodology – the so-called *enthalpy damping* – was suggested by Jameson for the solution of the potential flow equation [3]. It employs the difference between the total enthalpy H and its freestream value H_∞ to define an additional forcing term. With this, the convergence to the steady state can be considerably accelerated. The application of the enthalpy damping to the Euler equations (2.45) results in the following modification [4]

$$\frac{\partial}{\partial t}\int_\Omega \vec{W}\,d\Omega + \oint_{\partial\Omega} \vec{F}_c\,dS = \int_\Omega (\vec{Q} - \vec{Q}_{ED})\,d\Omega\,, \tag{9.1}$$

where the forcing term $\vec{Q}_{ED}$ is given by

$$\vec{Q}_{ED} = \vartheta \begin{bmatrix} \rho\,(H - H_\infty) \\ \rho u\,(H - H_\infty) \\ \rho v\,(H - H_\infty) \\ \rho w\,(H - H_\infty) \\ \rho\,(H - H_\infty) \end{bmatrix}. \tag{9.2}$$

The damping factor ϑ is a small constant, which has to determined empirically. The spatial discretisation scheme, and particularly the artificial dissipation has to be implemented in such a way that $H = H_\infty$ is a valid solution of the discretised equations. Then, the addition of the source term $\vec{Q}_{ED}$ does not alter the final steady state.

The enthalpy damping is conducted as an additional step after each update of the flow solution. For example, in the case of the explicit m-stage time-stepping scheme (Subsection 6.1.1 or 6.1.2), the damping step reads

$$\vec{W}_I^{n+1} = \frac{1}{1 + \vartheta(H_I^{(m)} - H_\infty)}\,\vec{W}_I^{(m)} \tag{9.3}$$

with the exception of the energy equation which becomes

$$(\rho E)_I^{n+1} = \frac{1}{1+\vartheta}\left[(\rho E)_I^{(m)} - \vartheta p_I^{(m)}\right]. \tag{9.4}$$

In Eq. (9.3), $\vec{W}_I^{(m)}$ denotes the final solution of the m-stage explicit time-stepping scheme. Numerical experiments in Ref. [1], which were conducted for a transonic flow past the NACA 0012 airfoil ($M_\infty = 0.8$, $\alpha = 1.25°$), confirmed that the number of time steps to reach the steady state can be reduced by about factor of two.

9.3 Residual Smoothing

The maximum CFL number and the convergence properties of the explicit multistage time-stepping scheme (Subsections 6.1.1 and 6.1.2) can be influenced by optimising the stage coefficients [5]-[7]. Jameson and Baker [8] introduced the *residual smoothing* technique with the aim to lend the explicit scheme an implicit character and hence to increase the maximum allowable CFL number. A further purpose of the residual smoothing is a better damping of the high-frequency error components of the residual. This is of particular importance for a successful application of the multigrid method.

The residual smoothing can be implemented in explicit, implicit or in mixed manner [9], [10], respectively. The residual smoothing is usually applied in each stage of the explicit time-stepping scheme (Eqs. (6.5), (6.7)). The previously computed residuals $\vec{R}^{(k)}$ are replaced by the smoothed residuals $\vec{R}^*$ before the solution $\vec{W}^{(k)}$ is updated. In the following, we shall discuss the implementation of the popular *Implicit Residual Smoothing* (IRS) on structured as well as on unstructured grids.

9.3.1 Central IRS on Structured Grids

The standard formulation of the implicit residual smoothing reads in 3D

$$\begin{aligned}
-\epsilon^I \vec{R}^*_{I-1,J,K} + (1+2\epsilon^I)\vec{R}^*_{I,J,K} - \epsilon^I \vec{R}^*_{I+1,J,K} &= \vec{R}_{I,J,K} \\
-\epsilon^J \vec{R}^{**}_{I,J-1,K} + (1+2\epsilon^J)\vec{R}^{**}_{I,J,K} - \epsilon^J \vec{R}^{**}_{I,J+1,K} &= \vec{R}^*_{I,J,K} \\
-\epsilon^K \vec{R}^{***}_{I,J,K-1} + (1+2\epsilon^K)\vec{R}^{***}_{I,J,K} - \epsilon^K \vec{R}^{***}_{I,J,K+1} &= \vec{R}^{**}_{I,J,K}\,,
\end{aligned} \tag{9.5}$$

where $\vec{R}^*_{I,J,K}$, $\vec{R}^{**}_{I,J,K}$, and $\vec{R}^{***}_{I,J,K}$ denote the smoothed residuals in I-, J-, and K-direction, respectively. The parameters ϵ^I, ϵ^J, and ϵ^K stand for the smoothing coefficients in the three computational coordinates. The implicit operator in Eq. (9.5) resembles second-order central difference. The term *Central Implicit Residual Smoothing* (CIRS) is therefore used. The implicit system in Eq. (9.5) is solved by the Thomas algorithm for the inversion of tridiagonal matrices.

The smoothing coefficients are usually defined as functions of spectral radii of the convective flux Jacobians [11]. The purpose is to apply only as much smoothing in each coordinate direction as it is necessary for stability and good error damping. A suitable formula for 2D was suggested in [12]

$$\begin{aligned}
\epsilon^I &= \max\left\{\frac{1}{4}\left[\left(\frac{\sigma^*}{\sigma}\,\frac{1}{1+\Psi r}\right)^2 - 1\right], 0\right\} \\
\epsilon^J &= \max\left\{\frac{1}{4}\left[\left(\frac{\sigma^*}{\sigma}\,\frac{1}{1+\Psi/r}\right)^2 - 1\right], 0\right\}.
\end{aligned} \tag{9.6}$$

Here, σ^*/σ denotes the ratio of the CFL numbers of the smoothed and unsmoothed scheme. The variable r stands for the ratio of the convective spectral

radii (Eq. (4.53)), i.e., $r = \hat{\Lambda}_c^J/\hat{\Lambda}_c^I$. The parameter $\Psi \approx 0.125$ ensures linear stability of the smoothing operation.

In 3D, the smoothing coefficients can be evaluated using an expression similar to Eq. (9.6) [12]

$$\begin{aligned}
\epsilon^I &= \max\left\{\frac{1}{4}\left[\left(\frac{\sigma^*}{\sigma}\frac{1}{1+\Psi(r^{JI}+r^{KI})}\right)^2 - 1\right], 0\right\} \\
\epsilon^J &= \max\left\{\frac{1}{4}\left[\left(\frac{\sigma^*}{\sigma}\frac{1}{1+\Psi(r^{KJ}+r^{IJ})}\right)^2 - 1\right], 0\right\} \\
\epsilon^K &= \max\left\{\frac{1}{4}\left[\left(\frac{\sigma^*}{\sigma}\frac{1}{1+\Psi(r^{IK}+r^{JK})}\right)^2 - 1\right], 0\right\},
\end{aligned} \tag{9.7}$$

where

$$r^{JI} = \hat{\Lambda}_c^J/\hat{\Lambda}_c^I, \quad r^{KI} = \hat{\Lambda}_c^K/\hat{\Lambda}_c^I,$$

etc. The typical value of the parameter Ψ is 0.0625.

The maximum of the ratio of the CFL numbers σ^*/σ depends on the value of the smoothing coefficient and on the type of the spatial discretisation scheme. In the case of the central scheme (Subsection 4.3.1), the ratio is given by

$$\frac{\sigma^*}{\sigma} \leq \sqrt{1+4\epsilon}\,. \tag{9.8}$$

In practice, value of $\sigma^*/\sigma \approx 2$ can be reached ($\epsilon = 0.8$). Higher ratios reduce the damping of the time-stepping scheme. There is no such simple condition like (9.8) for the upwind spatial discretisation (the maximum of σ^*/σ depends also on the stage coefficients). However, the CFL number (see Tables 6.1 and 6.2) can also be approximately doubled.

The limitation of the time step due to the viscous spectral radius $\hat{\Lambda}_v$ in Eq. (6.18) can be offset with the aid of the implicit residual smoothing. The maximum time step is calculated according to the formula (6.14) without $\hat{\Lambda}_v$. In flow regions where the viscous spectral radius dominates, smoothing is carried out using higher coefficients ϵ^I, ϵ^J, ϵ^K. Smoothing coefficients based on the viscous spectral radii can be calculated from [12], [13]

$$\begin{aligned}
\epsilon_v^I &= C\left(\frac{\sigma^*}{\sigma}\right)\frac{\hat{\Lambda}_v^I}{\hat{\Lambda}_c^I+\hat{\Lambda}_c^J+\hat{\Lambda}_c^K} \\
\epsilon_v^J &= C\left(\frac{\sigma^*}{\sigma}\right)\frac{\hat{\Lambda}_v^J}{\hat{\Lambda}_c^I+\hat{\Lambda}_c^J+\hat{\Lambda}_c^K} \\
\epsilon_v^K &= C\left(\frac{\sigma^*}{\sigma}\right)\frac{\hat{\Lambda}_v^K}{\hat{\Lambda}_c^I+\hat{\Lambda}_c^J+\hat{\Lambda}_c^K},
\end{aligned} \tag{9.9}$$

where the constant $C \approx 5/4$. The maximum of the coefficients from Eqs. (9.7) and (9.9) is then taken as the resulting smoothing coefficient.

9.3.2 Central IRS on Unstructured Grids

CIRS is implemented on unstructured grids by applying the Laplacian operator (see Eq. (5.20)) to the residual. Thus, the smoothed residual $\vec{R}_I^*$ in a control volume I is obtained from the implicit relation [14]

$$\vec{R}_I^* + \sum_{J=1}^{N_A} \epsilon \left(\vec{R}_I^* - \vec{R}_J^*\right) = \vec{R}_I \,. \tag{9.10}$$

The sum includes all N_A adjacent control volumes. The relation (9.10) is solved for $\vec{R}_I^*$ using Jacobi iteration. Useful values of the smoothing coefficient are $0.5 \leq \epsilon \leq 0.8$. With this values of ϵ it is possible to double the CFL number (and hence the time step). Due to the diagonal dominance of the matrix, the Jacobi iteration converges in about two steps.

9.3.3 Upwind IRS on Structured Grids

The previously discussed CIRS method works satisfactorily for subsonic and transonic flows. It is also helpful in viscous dominated regions. However, CIRS exhibits poor error-damping characteristics in conjuction with upwind spatial discretisation schemes. Furthermore, the robustness of a multistage scheme accelerated by CIRS suffers in the case of strong shocks. Therefore, the so-called *Upwind Implicit Residual Smoothing* (UIRS) method was developed [15], [16], which is particularly suited for high Mach-number flows.

By contrast to CIRS, the UIRS methodology takes the sign of the convective eigenvalues $\bar{\Lambda}_c$ into account. The idea is to smooth the residuals only in the direction of the characteristic of the Euler equations (i.e., dx/dt = const. = Λ_c). This approach prevents unphysical influences on the upstream residuals. The UIRS method requires transformation of the residuals into the characteristic variables (cf. Appendix A.9). In this way, each component of the residual vector can be smoothed independently according to the sign of the corresponding eigenvalue. For the l-th component of the residual, the implicit operator is defined in 1D as [15], [16]

$$\begin{aligned} -\epsilon^I (R^*)^l_{I-1} + (1+\epsilon^I)(R^*)^l_I \qquad\qquad &= (R^c)^l_I \qquad \text{if} \quad \Lambda^l_c > 0 \\ (1+\epsilon^I)(R^*)^l_I - \epsilon^I (R^*)^l_{I+1} &= (R^c)^l_I \qquad \text{if} \quad \Lambda^l_c < 0 \,, \end{aligned} \tag{9.11}$$

where $\vec{R}^c$ denotes the residual transformed into characteristic variables, i.e.,

$$\vec{R}^c = \bar{T}^{-1} \vec{R} \,. \tag{9.12}$$

The smoothed residuals $\vec{R}^*$ are obtained by the solution of a tridiagonal (bidiagonal if Λ^l_c does not change its sign) equation system using the Thomas algorithm. Afterwards, the residuals $\vec{R}^*$ are transformed back into the conservative variables and the solution can be updated.

The most desirable feature of UIRS is that it leads to very favourable damping properties of the multistage scheme, particularly in conjuction with an upwind spatial discretisation. The ability to damp solution errors remains or even improves for high smoothing coefficients. It was demonstrated for 1-D Euler equations that values like $\epsilon = 500$ and $\sigma^* = 1000$ result in a stable and very fast explicit multistage scheme [15], [16]. However, the problem is the implementation of UIRS in multiple dimensions. Since the convective flux Jacobians cannot be diagonalised simultaneously in all coordinate directions (see Appendix A.9), the transformation Eq. (9.12) and the smoothing Eq. (9.11) have to be carried out separately for each computational coordinate. The effect of the coordinate splitting is a reduced maximum smoothing coefficient to $2 \leq \epsilon \leq 6$. Despite this, the convergence to the steady state is strongly accelerated as compared to CIRS [17], [16]. The largest improvements in terms of convergence speed and robustness occur in combination with multigrid [18], [16].

The smoothing coefficients in multiple dimensions are scaled by the eigenvalues. For example, in 2D the following formula can be employed

$$\begin{aligned}
\epsilon^I &= \epsilon \cdot \min\left[\frac{\Lambda_c^I}{\Lambda_c^J}, 1\right] \\
\epsilon^J &= \epsilon \cdot \min\left[\frac{\Lambda_c^J}{\Lambda_c^I}, 1\right] .
\end{aligned} \tag{9.13}$$

The relation between the CFL number of the smoothed scheme and the coefficient ϵ reads

$$\frac{\sigma^*}{\sigma} \leq 1 + C\epsilon . \tag{9.14}$$

The constant C depends on the kind of the spatial discretisation. In the case of the central scheme $C = 1$. For the 1st- or 2nd-order upwind scheme the value is $C = 2$.

In order to circumvent the numerical effort of the transformation to the characteristic variables, a simplified version of the UIRS method was suggested [17], [18], [16]. Written in the I-direction it becomes

$$\begin{aligned}
-\epsilon^I \vec{R}^*_{I-1} + (1+\epsilon^I)\vec{R}^*_I &= \vec{R}_I \qquad \text{if} \quad M^I > 1 \\
-\epsilon^I \vec{R}^*_{I-1} + (1+2\epsilon^I)\vec{R}^*_I - \epsilon^I \vec{R}^*_{I+1} &= \vec{R}_I \qquad \text{if} \quad |M^I| < 1 \\
(1+\epsilon^I)\vec{R}^*_I - \epsilon^I \vec{R}^*_{I+1} &= \vec{R}_I \qquad \text{if} \quad M^I < -1 .
\end{aligned} \tag{9.15}$$

The Mach number M is based on velocity projected into the direction of the particular computational coordinate (here: I). Due to the low operation count, the simplified UIRS is especially suitable for 3-D flow problems. In Ref. [16] it was demonstrated that the CPU time needed to reach the steady state can be halved as compared to CIRS (hypersonic flow past a blunt cylinder, $M_\infty = 8$).

9.4 Multigrid

The multigrid methodology is a very powerful acceleration technique. It is based on the solution of the governing equations on a series of successively coarser grids. The solution updates from the coarse grid are then combined and added to the solution on the finest grid. The technique was originally developed by Brandt [19] for elliptic partial differential equations and later applied to the Euler equations by Jameson [20]-[22]. After that, the multigrid scheme was employed to solve the Navier-Stokes equations [11]-[13], [23]-[31]. The multigrid method can be implemented for both the explicit and the implicit time-stepping schemes [32]-[36]. The goal of the current research is the significant improvement of the efficiency of multigrid for hyperbolic flow problems [37], [38].

The basic idea of the multigrid scheme is to employ coarse grids in order to drive the solution on the finest grid faster to steady-state. Two effects are utilised for this purpose:

1. larger time steps can be employed on the coarser grids (owing to a larger control volume) in conjuction with a reduced numerical effort. Since the work for determining a new solution is distributed mainly over the coarser grids, a more rapid convergence and a reduction of the computing time results.

2. The majority of the explicit and implicit time-stepping and iterative schemes reduces efficiently mainly the high-frequency components of the solution error (see Section 10.3). The low-frequency components are usually only hardly damped. This results in a slow convergence to the steady state, after the initial phase (where the largest errors are eliminated) is over. The multigrid scheme helps at this point – the low-frequency components on the finest grid becomes high-frequency components on the coarser grids and are successively damped. As a result, the entire error is very quickly reduced, and the convergence is significantly accelerated.

Thus, as we can see, the success of the multigrid scheme depends heavily on good damping of the high-frequency error components by the time-stepping or iterative scheme.

An alternative to the geometrical multigrid is provided by the *Algebraic Multigrid* (AMG) method [39]-[43]. The AMG technique was developed for implicit schemes, where it operates directly on the system matrix (the left-hand side operator). The basic idea of AMG is to apply a coarsening matrix in order to reduce the dimension of the implicit operator and hence the number of equations. The reduced system, which represents a coarse level, is then solved to obtain the correction of the fine-level solution. The coarsening matrix is constructed such that the equations with the strongest coupling (i.e., the largest off-diagonals in the system matrix) are added together. Thus, the generation of coarse levels is governed solely by the physics of the flow problem and not by the grid. Therefore, the advantage of AMG is that no coarse grid topology has to be constructed or stored, which is particularly beneficial on unstructured grids.

9.4.1 Basic Multigrid Cycle

Before the geometric multigrid scheme can be applied, the coarser grids have to be generated. The standard way is to coarsen the grid evenly in all coordinate directions. However, Mulder [44] proposed an approach called *semicoarsening*. Here, the grid is coarsened only in one direction, which is changed from one coarse level to another. The semicoarsening methodology is especially suited for flow problems, where the governing equations are stiff in one spatial direction. An example is the direction normal to the wall in boundary layers. Applications of semicoarsening to the Navier-Stokes equations were reported, e.g., in Refs. [26], [45], [46].

The discretised governing equations on the fine grid read in accordance with the relationship (6.1)

$$\frac{d}{dt}\vec{W}_h = -\frac{1}{\Omega_h}\vec{R}_h\,. \tag{9.16}$$

In the following, the finest grid will be denoted by subscript h in reference to the spacing of the grid lines (characteristic dimension of the control volume on unstructured grids). The result, starting from a known solution $\vec{W}_h^n$, is a new solution $\vec{W}_h^{n+1}$ after one time step with some suitable iterative scheme. A new residual $\vec{R}_h^{n+1}$ is evaluated with this solution. In order to improve the solution $\vec{W}_h^{n+1}$ using a coarse grid, the following three steps are carried out:

1. Transfer of the Solution and Residuals to the Coarser Grid

The solution is transferred to the coarse grid by means of interpolation

$$\vec{W}_{2h}^{(0)} = \hat{I}_h^{2h}\,\vec{W}_h^{n+1}\,, \tag{9.17}$$

where the subscript $2h$ denotes the coarse grid [1] and $\hat{I}_h^{2h}$ is the *interpolation operator*. The residuals have to be transferred to the coarse grid as well, so that their low-frequency error components can be smoothed. A conservative transfer operator is employed for this purpose. This means when the control volume size increases, the value of the residual must increase by the same amount. The residuals of the fine grid are also required in order to retain the solution accuracy of the fine grid on the coarse grid. For this purpose, a source term, the so-called *forcing function* [19], [22], is formed as the difference between the residual transferred from the fine grid and the residual calculated using the initial solution $\vec{W}_{2h}^{(0)}$ (Eq. (9.17)) on the coarse grid, i.e,

$$(\vec{Q}_F)_{2h} = I_h^{2h}\vec{R}_h^{n+1} - \vec{R}_{2h}^{(0)}\,. \tag{9.18}$$

Here, I_h^{2h} represents the *restriction operator* which transfers residuals from the fine to the coarse grid. This type of multigrid scheme is known as the *Full Approximation Storage* (FAS) method [19]. The FAS method is particularly suited

[1] The notation $2h$ must not be understood in strictly geometric sense. On unstructured grids, the ratio of the characteristic dimensions of the control volumes on the fine and the coarse grid will usually differ from two. The same holds also for semicoarsening

for non-linear equations because the nonlinearities in the system are carried down to the coarse levels through the re-discretisation.

2. Calculation of a New Solution on the Coarse Grid

The solution is evaluated on the coarse grid in the same way as on the fine grid. The forcing function in Eq. (9.18) is added to the residual of the coarse grid, i.e.,

$$(\vec{R}_F)_{2h} = \vec{R}_{2h} + (\vec{Q}_F)_{2h} \,. \tag{9.19}$$

Hence, the time-stepping scheme can be written in the form

$$\frac{d}{dt}\vec{W}_{2h} = -\frac{1}{\Omega_{2h}}(\vec{R}_F)_{2h} = -\frac{1}{\Omega_{2h}}\left[\vec{R}_{2h} + (\vec{Q}_F)_{2h}\right] \,. \tag{9.20}$$

In the case of the explicit multistage scheme (Subsection 6.1.1), this results in

$$\begin{aligned} \vec{W}_{2h}^{(k)} &= \vec{W}_{2h}^{(0)} - \alpha_k \frac{\Delta t_{2h}}{\Omega_{2h}}\left[\vec{R}_{2h}^{(k-1)} + (\vec{Q}_F)_{2h}\right], \quad k = 1, \cdots, m \\ \vec{W}_{2h}^{n+1} &= \vec{W}_{2h}^{(m)} \end{aligned} \tag{9.21}$$

in accordance with Eq. (6.5). It has to be noted that during the first iteration (stage in Eq. (9.21)), $(\vec{R}_F)_{2h}$ is identical to the residual transferred from the fine grid (i.e., $(\vec{R}_F)_{2h} = I_h^{2h}\vec{R}_h^{n+1}$ from Eq. (9.18)). This guarantees that the solution on the coarse grid depends on the residual of the fine grid and thus retains the accuracy of the fine grid.

An important question is the accuracy of the spatial discretisation scheme on coarse grids. Since the coarse grids do not influence the accuracy of the fine-grid solution, first-order schemes are sufficient. The advantages of first-order accurate discretisation on coarse grids are the increased robustness, better damping properties, and lower numerical effort in comparison to higher-order schemes.

3. Solution Interpolation from the Coarse to the Fine Grid

After one or several time steps (iterations) were carried out on the coarse grid, the correction with respect to the initial – interpolated – solution (Eq. (9.17)) is calculated. This so-called *coarse grid correction* is given by

$$\delta\vec{W}_{2h} = \vec{W}_{2h}^{n+1} - \vec{W}_{2h}^{(0)} \,. \tag{9.22}$$

The coarse-grid correction is interpolated to the fine grid in order to improve the solution there. Hence, the new solution on the fine grid reads

$$\vec{W}_h^{+} = \vec{W}_h^{n+1} + I_{2h}^{h}\delta\vec{W}_{2h} \,, \tag{9.23}$$

where I_{2h}^h is denoted as the *prolongation operator*.

9.4.2 Multigrid Strategies

The basic multigrid scheme described above consists of one coarse grid only. If multiple coarse grids are present, steps 1 and 2 are repeated until the coarsest grid is reached. It is important to realize that the forcing function on the coarse grids is formed from the restricted **corrected** residual of Eq. (9.19)). For example, on the coarse grid $4h$, the forcing function is obtained from

$$(\vec{Q}_F)_{4h} = I_{2h}^{4h}(\vec{R}_F)_{2h}^{n+1} - \vec{R}_{4h}^{(0)} = I_{2h}^{4h}\left[\vec{R}_{2h}^{n+1} + (\vec{Q}_F)_{2h}\right] - \vec{R}_{4h}^{(0)}\,. \tag{9.24}$$

In this way, the residual of the finest grid controls the accuracy of the solution on all coarse grids. After a given number of time steps on the coarsest grid, step 3 can be successively repeated until the finest grid is reached again. This procedure is known as a saw-tooth or V-cycle (see Fig. 9.1a). However, it is also possible to conduct more cycles on the coarse grids. This strategy termed the W-cycle is displayed in Fig. 9.1b. It is employed particularly frequently for transonic flows. In the case of supersonic and hypersonic flows, the V-cycle proved to be more efficient.

Number of Time Steps

The optimum number of time steps before the restriction and after the prolongation depends on the type of the time-stepping scheme. In the case of the explicit multistage scheme (Subsection 6.1.1 or 6.1.2), it is common to carry out only one time step before the restriction of residuals and no time step after the prolongation. However, the robustness of the multigrid scheme can be improved by smoothing the coarse grid corrections (Eq. (9.22)) before adding them to the fine grid solution $\vec{W}_h^{n+1}$ (Eq. (9.23)). The same central implicit smoothing (with constant coefficients) as described in Section 9.3 is utilised.

The other popular time-stepping method, the implicit LU-SGS scheme (see Subsection 6.2.4), requires two iterations before the restriction for the best multigrid efficiency [34]. The number of time steps after the prolongation depends on the spatial discretisation. In the case of the central scheme (Subsection 4.3.1), no time step is necessary [34]-[36], but the solution correction can be smoothed. On contrary, one time step should carried out after the prolongation if an upwind spatial discretisation is used. This (2,1)- strategy proved to be an optimum with respect to robustness and computing time for various flow conditions [35], [36].

Starting Grid

It should be pointed out that in practice the multigrid scheme is not started directly from the finest grid. Instead, several multigrid cycles are executed from one of the coarse grids. The approximate solution is interpolated to the next finer grid (using the same operator as for the prolongation), several cycles are executed and so on, until the finest grid is reached. In this way, a good starting

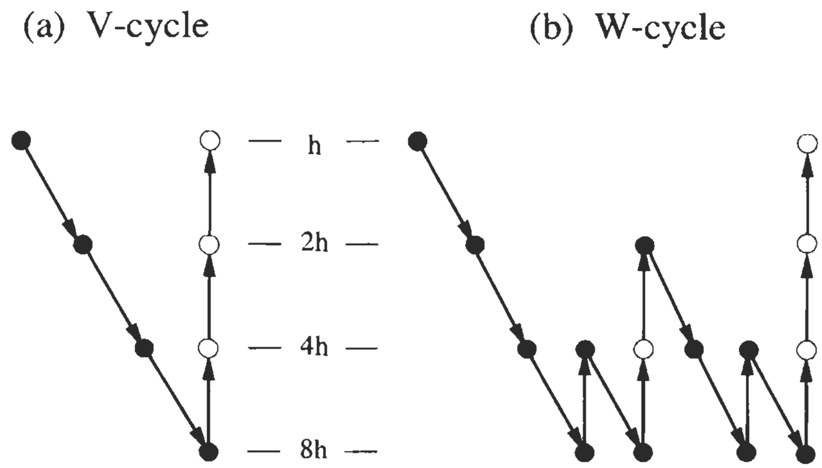


Figure 9.1: Types of multigrid cycles. • denotes time steps before restriction; ∘ represents time steps after prolongation.

solution is obtained on the finest grid with only a low numerical effort. This very efficient procedure is termed the *Full Multigrid* (FMG) method [19].

Accuracy of Transfer Operators

The restriction (9.18) and the prolongation (9.23) operator must fulfil certain accuracy requirements, namely [47]

$$m_R + m_P > m_E \,, \tag{9.25}$$

where m_R and m_P denote the degree plus 1 of the polynomial, which is exactly interpolated by the restriction and the prolongation operator, respectively. For example, m_R or m_P are equal to two in the case of linear interpolation. Furthermore, m_E represents the order of the governing equations. Thus, $m_E = 1$ for the Euler equations, and $m_E = 2$ in the case of the Navier-Stokes equations. If the condition (9.25) is violated, the additional errors introduced by the restriction and/or prolongation will disturb the fine-grid solution. Hence, such multigrid scheme will converge slowly or it will even diverge.

9.4.3 Implementation on Structured Grids

The implementation of multigrid on structured grids is straightforward since the coarse grids can be easily generated by deleting every second grid line in the respective coordinate direction. The spacing of the grid lines is therefore $2h, 4h$, etc. This guarantees that the numerical effort on the coarse grids stays low as compared to the finest grid. Several representative examples are provided in Refs. [11]-[13], [20]-[22], [26], [32]-[36].

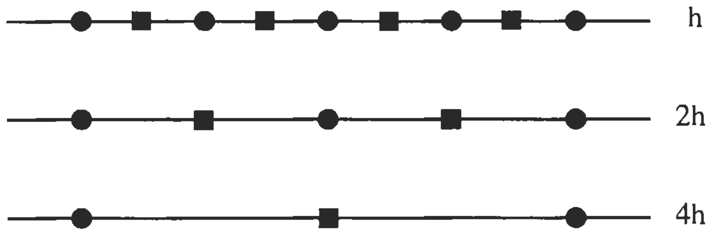


Figure 9.2: Representation of a fine (h) and of two coarse ($2h, 4h$) 1-D grids. Circles denote grid points, rectangles represent cell centres.

As we can conclude from Fig. 9.2, the operators for the solution interpolation, the restriction of residuals and the prolongation of corrections have to be defined differently for cell-centred and node-centred (cell-vertex) schemes. For example, two successive grids have every second grid point in common. On contrary, the cell centres are always at different locations. Therefore, we shall discuss the standard forms of the transfer operators separately for the cell-centred and node-centred (identical to cell-vertex) finite-volume schemes.

Apart from the symmetrical, purely geometrically defined restriction and prolongation operators, which will be presented next, upwind-biased forms were suggested in [16], Chapter 4. The upwind restriction and prolongation, which accounts for the characteristics of the flow equations, improve the robustness of the multigrid scheme for hypersonic flows. In order to save space, we shall discuss the implementation of an upwind prolongation operator for the node-centred discretisation scheme only.

Transfer Operators for the Cell-Centred Scheme

The solution is transfered from the fine to the coarse grid by using a volume weighted interpolation. In 2D, Eq. (9.17) becomes (see Fig. 9.3a)

$$(\vec{W}_{2h}^{(0)})_{I,J} = \frac{(\vec{W}_h^{n+1})_{I,J}\Omega_{I,J} + (\vec{W}_h^{n+1})_{I+1,J}\Omega_{I+1,J} + (\vec{W}_h^{n+1})_{I,J+1}\Omega_{I,J+1}}{\Omega_{I,J} + \Omega_{I+1,J} + \Omega_{I,J+1} + \Omega_{I+1,J+1}} + \frac{(\vec{W}_h^{n+1})_{I+1,J+1}\Omega_{I+1,J+1}}{\Omega_{I,J} + \Omega_{I+1,J} + \Omega_{I,J+1} + \Omega_{I+1,J+1}}\,. \tag{9.26}$$

Similar transfer operator is employed in 3D, where the summation is over the eight fine-grid control volumes, which form one coarse-grid cell.

The restriction operator is defined as a sum of the residuals from all cells which are contained in one coarse-grid control volume. Hence, in 2D we have (cf. Fig. 9.3a)

$$(I_h^{2h}\vec{R}_h^{n+1})_{I,J} = (\vec{R}_h^{n+1})_{I,J} + (\vec{R}_h^{n+1})_{I+1,J} + (\vec{R}_h^{n+1})_{I,J+1} + (\vec{R}_h^{n+1})_{I+1,J+1} \tag{9.27}$$

and likewise in 3D. Those residuals $\vec{R}_h^{n+1}$ in Eq. (9.27), which are located outside the physical domain, should be set to zero.

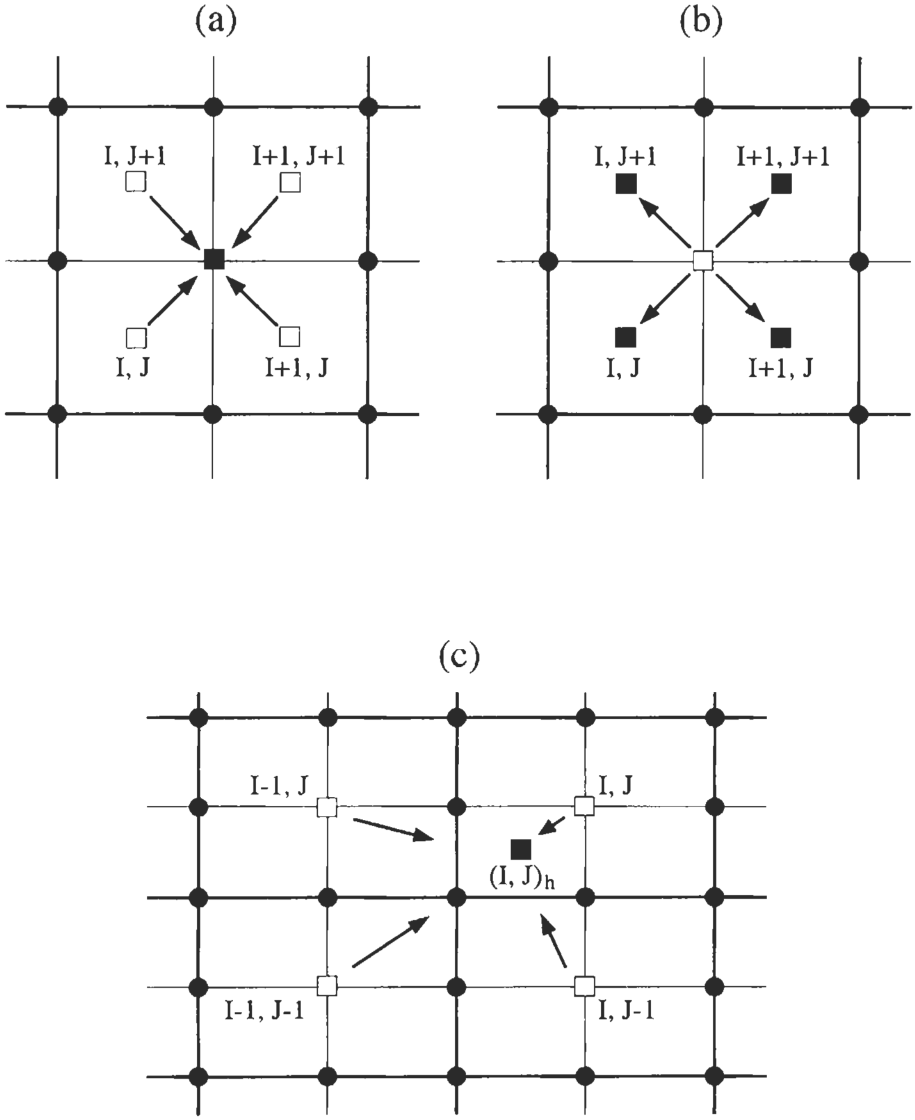


Figure 9.3: Solution interpolation and residual restriction (a), prolongation of coarse-grid correction (b, c) for a structured cell-centred scheme in 2D. Filled circle = grid point; filled rectangle = cell centre **to** which it is interpolated; rectangle = cell centre **from** which it is interpolated; thick line = coarse grid; thin line = fine grid.

The prolongation of the coarse-grid correction Eq. (9.23) can be conducted in two different ways. First, a zeroth-order prolongation operator results, if $\delta\vec{W}_{2h}$ is equally distributed to all surrounding cell centres as indicated in Fig. 9.3b. Thus, for instance

$$(\vec{W}_h^+)_{I,J+1} = (\vec{W}_h^{n+1})_{I,J+1} + (\delta\vec{W}_{2h})_{I,J}\,, \quad \text{etc.} \tag{9.28}$$

The second possibility, which leads to a faster convergence of the multigrid scheme, consists of two steps. In a first step, $\delta\vec{W}_{2h}$ is interpolated to the grid nodes like for the cell-vertex scheme (see below). In a second step, the nodal values are averaged to obtain the value in the centre of the fine-grid cell. Referring to Fig. 9.3c, the following final relationship can be derived

$$\begin{aligned}(I_{2h}^h\delta\vec{W}_{2h})_{I,J} = \frac{1}{16}\Big[&9(\delta\vec{W}_{2h})_{I,J} + 3(\delta\vec{W}_{2h})_{I-1,J} \\ &+3\,(\delta\vec{W}_{2h})_{I,J-1} + (\delta\vec{W}_{2h})_{I-1,J-1}\Big]\,.\end{aligned} \tag{9.29}$$

The corresponding expression in 3D can be found by a similar procedure. It reads

$$\begin{aligned}(I_{2h}^h\delta\vec{W}_{2h})_{I,J,K} = \frac{1}{64}\Big[&27(\delta\vec{W}_{2h})_{I,J,K} + 9(\delta\vec{W}_{2h})_{I-1,J,K} \\ &+9(\delta\vec{W}_{2h})_{I,J-1,K} + 9(\delta\vec{W}_{2h})_{I,J,K-1} \\ &+3(\delta\vec{W}_{2h})_{I-1,J-1,K} + 3(\delta\vec{W}_{2h})_{I-1,J,K-1} \\ &+3(\delta\vec{W}_{2h})_{I,J-1,K-1} + (\delta\vec{W}_{2h})_{I-1,J-1,K-1}\Big]\,.\end{aligned} \tag{9.30}$$

Transfer Operators for the Cell-Vertex Scheme

Because of the common nodes between the fine and coarse grid, the solution can be transfered simply by injection, i.e.,

$$(\vec{W}_{2h}^{(0)})_{i,j,k} = (\vec{W}_h^{n+1})_{i,j,k} \tag{9.31}$$

The standard central restriction operator represents a linear interpolation from the nodes of all four (eight in 3D) fine-grid cells, which resemble one coarse-grid cell. According to Fig. 9.4, the restricted residual is calculated in 2D as

$$\begin{aligned}(I_h^{2h}\vec{R}_h^{n+1})_{i,j} = (\vec{R}_h^{n+1})_{i,j} &+ \frac{1}{2}\Big[(\vec{R}_h^{n+1})_{i-1,j} + (\vec{R}_h^{n+1})_{i+1,j} \\ &+ (\vec{R}_h^{n+1})_{i,j-1} + (\vec{R}_h^{n+1})_{i,j+1}\Big] \\ &+ \frac{1}{4}\Big[(\vec{R}_h^{n+1})_{i-1,j-1} + (\vec{R}_h^{n+1})_{i+1,j-1} \\ &+ (\vec{R}_h^{n+1})_{i-1,j+1} + (\vec{R}_h^{n+1})_{i+1,j+1}\Big]\,.\end{aligned} \tag{9.32}$$

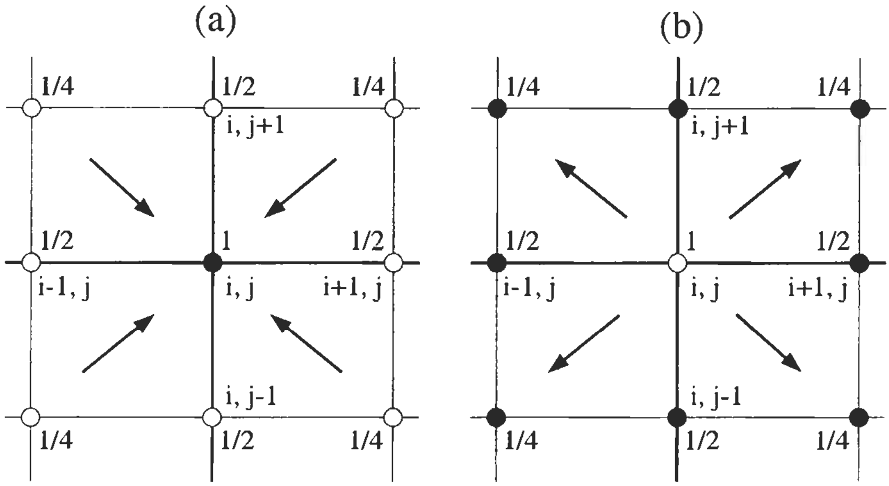


Figure 9.4: Interpolation factors in the case of the restriction (a) and the prolongation (b) for a structured cell-vertex scheme in 2D. Filled circle = point **to** which it is interpolated; circle = point **from** which it is interpolated; thick line = coarse grid; thin line = fine grid. Point $(i,\ j)$ is common to both grids.

In 3D, the fine-grid residuals are collected as follows

$$(I_h^{2h}\vec{R}_h^{n+1})_{i,j,k} = (\vec{R}_h^{n+1})_{i,j,k} + \frac{1}{2}\mathcal{A} + \frac{1}{4}\mathcal{B} + \frac{1}{8}\mathcal{C} \qquad (9.33)$$

with the factors

$$\begin{aligned}
\mathcal{A} = {} & (\vec{R}_h^{n+1})_{i+1} + (\vec{R}_h^{n+1})_{i-1} + (\vec{R}_h^{n+1})_{j+1} \\
& +(\vec{R}_h^{n+1})_{j-1} + (\vec{R}_h^{n+1})_{k+1} + (\vec{R}_h^{n+1})_{k-1} \\
\mathcal{B} = {} & (\vec{R}_h^{n+1})_{i+1,j+1} + (\vec{R}_h^{n+1})_{i-1,j+1} + (\vec{R}_h^{n+1})_{i+1,j-1} + (\vec{R}_h^{n+1})_{i-1,j-1} \\
& +(\vec{R}_h^{n+1})_{i+1,k+1} + (\vec{R}_h^{n+1})_{i-1,k+1} + (\vec{R}_h^{n+1})_{i+1,k-1} + (\vec{R}_h^{n+1})_{i-1,k-1} \\
& +(\vec{R}_h^{n+1})_{j+1,k+1} + (\vec{R}_h^{n+1})_{j-1,k+1} + (\vec{R}_h^{n+1})_{j+1,k-1} + (\vec{R}_h^{n+1})_{j-1,k-1} \qquad (9.34) \\
\mathcal{C} = {} & (\vec{R}_h^{n+1})_{i+1,j+1,k+1} + (\vec{R}_h^{n+1})_{i-1,j+1,k+1} \\
& +(\vec{R}_h^{n+1})_{i-1,j-1,k+1} + (\vec{R}_h^{n+1})_{i+1,j-1,k+1} \\
& +(\vec{R}_h^{n+1})_{i+1,j+1,k-1} + (\vec{R}_h^{n+1})_{i-1,j+1,k-1} \\
& +(\vec{R}_h^{n+1})_{i-1,j-1,k-1} + (\vec{R}_h^{n+1})_{i+1,j-1,k-1}\ .
\end{aligned}$$

In the above Eq. (9.34), only those indices are shown which are different from

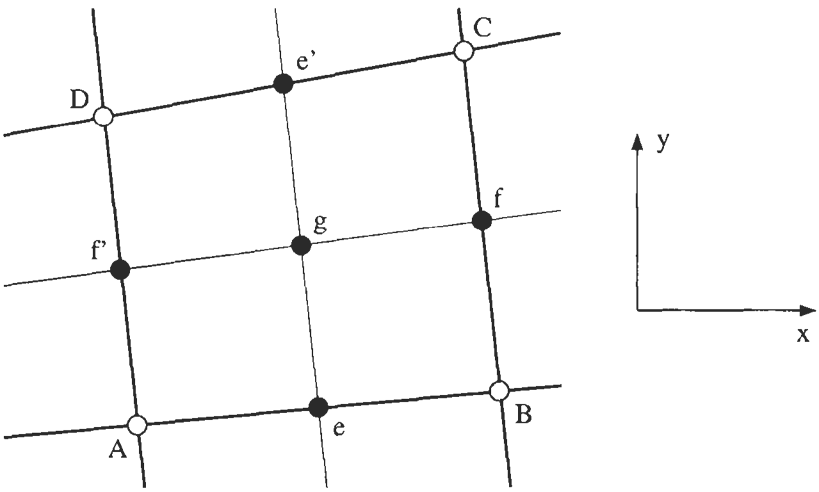


Figure 9.5: Upwind prolongation in 2D. Points A, B, C, D are common to the coarse (thick line) and the fine grid. Points e, f, e', f', g belong the fine grid (thin line) only.

i, j, k. It should be noted that the residuals $\vec{R}_h^{n+1}$ must be set to zero at all boundary points of the physical domain **before** the restriction.

The prolongation of the coarse-grid correction can be implemented as a loop over the points of the coarse grid. Within the loop, the values $(\delta\vec{W}_{2h})_{i,j,k}$ are distributed to the fine-grid points using the same weights as for the restriction (cf. Fig. 9.4b). The particular contributions are summed up in order to obtain the complete transfered correction at each point of the fine grid.

Upwind Prolongation (Cell-Vertex Scheme)

In principle, upwind prolongation can be formulated either in characteristic or in conservative variables. A particularly efficient implementation in conservative variables was proposed in [16]. The methodology employs upwind-biased interpolation of the corrections in Eq. (9.23) according to the Mach number and the velocity direction. The numerical effort is very low, nevertheless the robustness of the multigrid scheme can be significantly improved for high Mach-number flows [16].

The interpolation of the solution corrections $\delta\vec{W}_{2h}$ to the finer grid is accomplished in two steps. In the first step, the corrections at the points A, B, C, and D (see Fig. 9.5), which are common to both grids, are transferred directly to the finer grid. In the second step, the corrections are interpolated to the points e, f, e', f' and g, which are contained only on the finer grid. The interpolation depends on the sign and the absolute value of the Mach number at the corresponding point. The Mach number in this case is calculated using

the contravariant velocity. Thus, for example, at the point e the formula

$$M_e = \frac{\vec{n} \cdot \vec{v}_e}{c} . \tag{9.35}$$

is employed. The normal vector $\vec{n}$ in Eq. (9.35) can be obtained either by averaging the face vectors of the control volume (in the direction A–B), or by normalising the vector from point A to point B. Then, the correction is transferred to point e according to the rule

$$(I_{2h}^h \delta\vec{W}_{2h})_e = \begin{cases} (\delta\vec{W}_{2h})_A & \text{if} \quad M_e > 1 \\ \frac{1}{2}\left[(\delta\vec{W}_{2h})_A + (\delta\vec{W}_{2h})_B\right] & \text{if} \quad |M_e| \leq 1 \\ (\delta\vec{W}_{2h})_B & \text{if} \quad M_e < -1 \end{cases} . \tag{9.36}$$

The same procedure applies also to the points f to f'. The upwinding helps to match the information exchange between the grids better to the real physics. The interpolation to the point g is more difficult. In Ref. [16], the values at the surrounding points A to D were simply averaged

$$(I_{2h}^h \delta\vec{W}_{2h})_g = \frac{1}{4}\left[(\delta\vec{W}_{2h})_A + (\delta\vec{W}_{2h})_B + (\delta\vec{W}_{2h})_C + (\delta\vec{W}_{2h})_D\right] . \tag{9.37}$$

However, some sort of upwind weighted interpolation would be more appropriate. The upwind prolongation can be implemented in similar way also in 3D. Despite the simplification, encouraging results were obtained in a number of test cases [16].

9.4.4 Implementation on Unstructured Grids

As compared to the structured grids, the construction of the coarse grids is much more involved in the case of unstructured grids. The problem is how to construct an uniformly coarsened grid from a set of elements (grid cells) which have no particular ordering. Additionally, the ratio of the cell volumes of the coarse to the fine grid has to stay within a certain margins (about 4 in 2D and 8 in 3D). One possibility how to solve this problem is to apply the AMG methodology [39]-[43], which we briefly discussed at the beginning of Section 9.4. However, the geometric multigrid is still more widely used. Therefore, we shall concentrate here on this approach.

Three main methods for the generation of coarse grids can be identified:

- nonnested-grids approach,
- topological methods, and
- agglomeration of control volumes.

Reviews of the above methods were presented in Refs. [48] and [49].

The standard restriction and prolongation operators are based on purely geometrically defined interpolation. Leclerq and Stoufflet [50] suggested upwind transfer operators, which are particularly promising for flows with strong shocks. Their upwind restriction/prolongation is based on the transformation of the residuals/corrections into the characteristic variables. After an upwind-biased interpolation, the restricted/prolongated values are transformed back into the physical variables. Numerically much less expensive upwind multigrid method was presented in [16] (see also Subsection 9.4.3, Eq. (9.36)).

Nonnested Grids

The most obvious idea is to generate a sequence of completely independent, increasingly coarser grids [51]-[56]. It is not necessary that the grids contain any common nodes. Therefore, we speak of *nonnested* grids. However, it is important that the main geometrical features (leading and trailing edge, fuselage nose, etc.) are retained on all coarse grids. This is not easy to accomplish, particularly in the case of a geometrically complex configuration. Multigrid based on nonnested grids is hardly used today.

Topological Methods

One particular approach applies graph-based algorithms in order to remove certain nodes from the fine grid. The remaining nodes are then re-triangulated [57], [58]. On contrary to the nonnested-grids approach, the interpolation between the grids becomes easier, since the successive grids contain common nodes. However, the method inherits the drawback of the nonnested grids with respect to geometry conformance.

A further topological method employs *grid refinement* [59], [29], [60]. The technique starts from a coarse grid and generates finer grids by element division. The methodology can be applied either over the whole physical domain or only locally (e.g., at boundary layers). The disadvantage of this approach is that the quality of the finest grid strongly depends on the initial coarse grid and the refinement procedure. The problem can be partially cured by edge swapping [61], [62].

Another idea for the generation of coarse grids is based on *edge collapsing* [63]. It was initially developed for inviscid flows on tetrahedral grids. The edge-collapsing method was further extended to viscous flows on mixed-element grids in [64].

Agglomeration Multigrid Method

A very efficient methodology for unstructured grids is the so-called *agglomeration multigrid*. It was first presented by Lallemand [65], Lallemand et al. [66] and by Koobus et al. [67]. Later on, the agglomeration multigrid was adopted by various authors [68]-[72], [31]. The method generates a coarse grid by fusing the **control volumes** of the finer grid with their neighbours. The resulting

coarse grids consist of successively larger, irregularly shaped polyhedral cells. This is depicted in Fig. 9.6. As we can see, the agglomeration technique retains the full discretisation of the boundary surfaces. This represents a significant advantage over all previously discussed unstructured multigrid methods. However, it should be mentioned that up to now, implementations of the agglomeration multigrid were based mostly on the median-dual cell-vertex scheme (Subsection 5.2.2). The application of the agglomeration multigrid to a cell-centred scheme (Subsection 5.2.1) was described in [68].

Generation of Coarse Grids by Volume Agglomeration

The volume agglomeration for a node-centred scheme proceeds in the following steps:

1. build a list of the so-called *seed points*. Seed points are grid points selected to agglomerate the surrounding control volumes. The list of seed points can contain either those points which form an approximate maximal independent set [70], or simply all points of the current grid level.

2. Loop over all seed points.

3. If the seed point is unagglomerated, agglomerate all its nearest neighbours (connected by an edge), which were not already agglomerated.

4. Check the coarsening ratio (i.e., how many fine-grid control volumes are contained within a coarse-grid volume). If the ratio is less than four (eight in 3D), the neighbours of the already agglomerated nearest neighbours are added (if not associated with another seed point), until the optimum coarsening ratio is achieved. In Ref. [27], it was proposed to agglomerate those distance-two neighbours first, which are connected to at least two (three in 3D) agglomerated nearest neighbours.

5. If there are still seed points in the list, goto step 2.

6. Eliminate *singletons*. These are single control volumes which could not be agglomerated, because there were no unagglomerated neighbours. A singleton can be eliminated by agglomeration with such neighbouring control volume, which has the smallest coarsening ratio. This leads to coarse-grid levels with a more regular distribution of control-volume areas [31].

The above procedure is repeated until all coarse grids are generated.

The volume agglomeration has to start from the boundary in order to preserve grid isotropy. In 3D, user intervention may be required to prescribe the agglomeration direction depending on the shape of the boundary. To overcome this difficulty, Okamoto et al. [72] proposed another algorithm denoted as *global coarsening*. The method employs a global partitioning scheme, which is based on edge colouring. The partitioning scheme is used to generate an independent

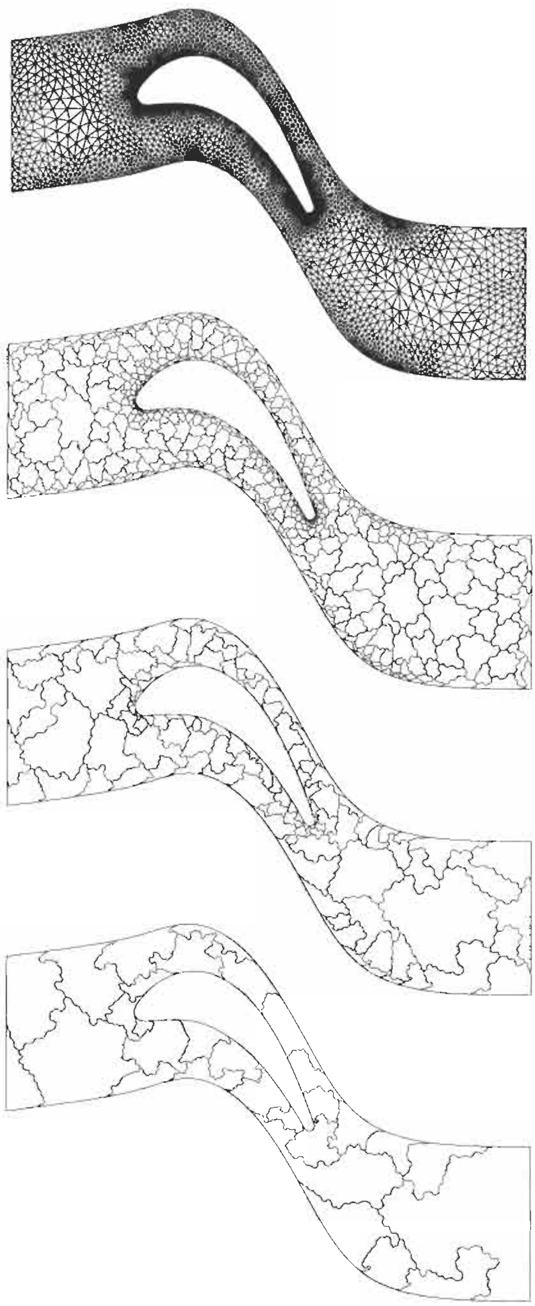


Figure 9.6: Generation of coarse grids by the agglomeration multigrid (median-dual scheme) in 2D. The sequence shows the finest grid and three coarse grids (top to bottom).

set of edges. In a second step, all control volumes, which share an edge of the independent set, are agglomerated. The procedure is repeated until the prescribed coarsening ratio is achieved. The method does not require the specification of an initial seed point or agglomeration direction. It can also treat any type of grid cells.

Problems of Agglomeration Multigrid

The implementation of agglomeration multigrid presents little difficulty for inviscid flows. The Euler equations are in general discretised as fluxes over individual control-volume faces. In this respect, it does not matter how complex is the shape of the control volume (in fact, averaged face vectors are used). Furthermore, a first-order accurate spatial scheme requires the knowledge of flow quantities in the neighbouring control volumes only. This information is readily available on the coarse grids.

In the case of viscous flows, the discretisation of the diffusive fluxes on arbitrary shaped control volumes is no longer straightforward. The problem is the evaluation of gradients at face midpoints (see the discussion in [31]). Another, and even more serious, difficulty is related to the required accuracy of the prolongation operator. The inequality in Eq. (9.25) suggests $m_P = 2$, since the common restriction operator (sum of residuals) leads to $m_R = 1$ only. However, the construction of a linear interpolation is not easy on the coarse grids.

In order to circumvent the construction of a first-order accurate prolongation operator, Mavriplis [48] proposed to use constant prolongation (i.e., all points of the fine grid contained within an agglomerated coarse-grid volume get the same solution correction) and a scaling of the viscous fluxes. However, this approach does not lead to optimum multigrid efficiency.

Haselbacher [31] suggested to retain the fine-grid discretisation of the viscous fluxes also on the coarse grids and to enforce the boundary conditions like on the finest grid. Moreover, he proposed a piecewise linear prolongation operator. For a scalar quantity U, it can be written as

$$I_{2h}^{h}\delta U_{2h} = (\delta U_{2h})_i + \Psi_i(\nabla\delta U_{2h})_i \cdot \vec{r}_{i,j}\,. \tag{9.38}$$

The gradient $(\nabla\delta U_{2h})_i$ in Eq. (9.38) is calculated by using the linear least-squares reconstruction described in Subsection 5.3.4, Eq. (5.51). The values of the limiter function Ψ_i are evaluated according to the Barth-Jespersen limiter function presented in Subsection 5.3.5.

9.5 Preconditioning for Low Mach Numbers

In the low subsonic Mach number regime, when the magnitude of the flow velocity becomes small in comparison with the acoustic speed, the convective terms of the governing equations (2.19) become stiff. We can demonstrate this with the following example. In the 3-D case, we have the five eigenvalues

$$\begin{aligned} (\Lambda_c)_{1,2,3} &= V && \text{(convective modes)} \\ (\Lambda_c)_{4,5} &= V \pm c && \text{(acoustic modes)}\,, \end{aligned} \tag{9.39}$$

where c denotes speed of sound. The stiffness of the governing equations (when marching in time) is determined by the characteristic *condition number*. This number is defined as the ratio of the largest to the smallest eigenvalue

$$C_N = \frac{|(\Lambda_c)_{max}|}{|(\Lambda_c)_{min}|} = \frac{|V| + c}{|V|} = \frac{M+1}{M}\,. \tag{9.40}$$

The allowable local time step is limited by the fastest moving wave, i.e, $(\Lambda_c)_4$. During one time step, the slowest wave moves only over a fraction of cell width: $\Lambda_{min}\Delta t \approx (\Lambda_{min}/\Lambda_{max})h = h/C_N$. Thus, a large condition number C_N (i.e., for $M \to 0$) reduces the efficiency of wave propagation – slows down the convergence to steady state [73]. Furthermore, it was demonstrated in [74], [75] that schemes for **compressible** flows have an amount of artificial dissipation which does not scale correctly for Mach number approaching zero. Thus, the accuracy of such spatial discretisation suffers at low Mach numbers [76].

If the velocity in the entire flow field is low ($M < 0.2$), than the compressibility effects can be neglected and the incompressible equations can be utilised. The incompressible Navier-Stokes equations can be solved by the well-known pressure-based schemes [77]. The other possibility is the application of the artificial compressibility (or pseudo-compressibility) method [78]-[83]. However, there are flow cases like the following ones:

- high-speed flows with large embedded regions of low velocity. An example is the subsonic flow upstream of a strongly converging nozzle.
- Low-speed flows that are compressible due to density changes induced by heat sources. This occurs for surface heat transfer or volumetric heat addition (combustion simulation).
- Problems, where compressible and incompressible flow at varying Mach numbers occur side by side – we speak of *all-speed flows*. Such situation arises, for instance, in propulsion, for high-lift configurations and in V/STOL manoeuvring.

Such cases require the application of the compressible governing equations. In order to solve them efficiently and accurately at low Mach numbers, *preconditioning* can be employed. The advantage of preconditioning is that it enables a solution method, which is applicable at all Mach numbers.

Preconditioning consists of the multiplication of the time-derivative term $(\partial \vec{W}/\partial t)$ by a matrix. The purpose of the ***preconditioning matrix*** is to equalise the eigenvalues as much as possible for $M \to 0$. In this way, the stiffness represented by the condition number Eq. (9.40) is reduced and the convergence of the time-stepping or iterative solution process is dramatically enhanced. The second effect of preconditioning is the change of the independent variables. Thus, the governing equations are solved for the pressure, the velocity components, and for the entropy or the temperature instead of $\vec{W} = [\rho, \rho u, \rho v, \rho w, \rho E]^T$. Therefore, the pressure becomes strongly coupled to the other equations, which is of great importance at low Mach numbers. The application of the preconditioning matrix (and/or the altered eigenvalues) to the numerical dissipation scheme [84] allows an accurate flow solution at vanishing Mach number.

The construction of a preconditioning matrix is relatively easy in the case of the Euler equations. The formulation proposed by van Leer at al. [84] achieves the lowest attainable condition number. The methodology was discussed in detail in [74]. However, the preconditioning of the Navier-Stokes equations is more involved. The reason is that the viscous terms lead to complex wave speeds, which makes the preconditioned system difficult to analyse. The most recognised preconditioners for viscous flows were proposed by Choi and Merkle [85], [86], Turkel [87]-[89], Lee and van Leer [90], [91] and Lee [75], Jorgenson and Pletcher [92], and by Weiss and Smith [93], [43], respectively. Examples of applications can be found in Refs. [94]-[98].

The preconditioned Navier-Stokes equations (2.19), formulated in primitive variables $\vec{W}_p = [p, u, v, w, T]^T$, can be written as

$$\bar{\Gamma}\,\frac{\partial}{\partial t}\int_{\Omega} \vec{W}_p \, d\Omega + \oint_{\partial\Omega} (\vec{F}_c - \vec{F}_v)\, dS = \int_{\Omega} \vec{Q}\, d\Omega \,, \tag{9.41}$$

where $\bar{\Gamma}$ denotes the preconditioning matrix. It is important to note that Eq. (9.41) is not conservative for unsteady flows. Therefore, the dual time-stepping approach (Section 6.3) has to be employed in order to obtain time-accurate solution.

The preconditioning matrix $\bar{\Gamma}$ due to Weiss and Smith [93], [43] was found to perform very well for various flow cases. It is defined as

$$\bar{\Gamma} = \begin{bmatrix} \theta & 0 & 0 & 0 & \rho_T \\ \theta u & \rho & 0 & 0 & \rho_T u \\ \theta v & 0 & \rho & 0 & \rho_T v \\ \theta w & 0 & 0 & \rho & \rho_T w \\ \theta H - \delta & \rho u & \rho v & \rho w & \rho_T H + \rho\, c_p \end{bmatrix}, \tag{9.42}$$

where

$$\rho_T = \left.\frac{\partial \rho}{\partial T}\right|_{p=\text{const.}} . \tag{9.43}$$

In the case of ideal gas, $\rho_T = -p/RT^2 = -\rho/T$. The parameter θ in Eq. (9.42) is given by

$$\theta = \frac{1}{u_r} - \frac{\rho_T}{\rho\, c_p} \tag{9.44}$$

and $\delta = 1$ ($\delta = 0$ for an incompressible fluid). The reference velocity u_r has to be defined such that the characteristic condition number remains bounded for $M \to 0$. In Ref. [43], it is suggested to determine u_r from the relation

$$u_r = \max\left(\|\vec{v}\|_2,\ \nu/\Delta x,\ \epsilon\sqrt{\Delta p/\rho}\right) \tag{9.45}$$

with Δx being the characteristic length of the control volume. Further, the quantity Δp represents the pressure difference between adjacent control volumes and ϵ is a small number ($\approx 10^{-3}$). As we can see from Eq. (9.45), u_r is not allowed to decrease below the local convective or diffusion velocity. In the case of compressible flows, the maximum value of u_r is limited by local speed of sound [93]. The pressure difference Δp in Eq. (9.45) is employed to stabilise the scheme near stagnation points.

The preconditioned Navier-Stokes equations (9.41) have the following convective eigenvalues [93]

$$\bar{\Lambda}_c = \bar{\Gamma}^{-1}\left(\frac{\partial \vec{F}_c}{\partial \vec{W}_p}\right) = [\,V,\ V,\ V,\ V' + c',\ V' - c'\,]\,, \tag{9.46}$$

where $V = \vec{v}\cdot\vec{n}$ denotes the contravariant velocity and

$$\begin{aligned} V' &= V(1-\alpha) \\ c' &= \sqrt{\alpha^2 V^2 + u_r^2} \\ \alpha &= \frac{1-\beta u_r^2}{2} \\ \beta &= \left(\rho_p + \frac{\rho_T}{\rho\, c_p}\right) \\ \rho_p &= \left.\frac{\partial \rho}{\partial p}\right|_{T=\text{const.}}\,. \end{aligned} \tag{9.47}$$

The changed eigenvalues and flux Jacobians have to be taken into account when implementing the spatial discretisation scheme. The effect on the central scheme with artificial dissipation (Subsection 4.3.1) is discussed, e.g., in [88]. The necessary changes to Roe's upwind scheme (Subsection 4.3.3) were described, e.g., in Ref. [93] or [99].

An example of the application of the preconditioning method Eq. (9.41) and (9.42) is presented in Fig. 9.7. We can clearly see that preconditioning not only helps to accelerate the convergence, but also to find the correct solution of the flow problem.

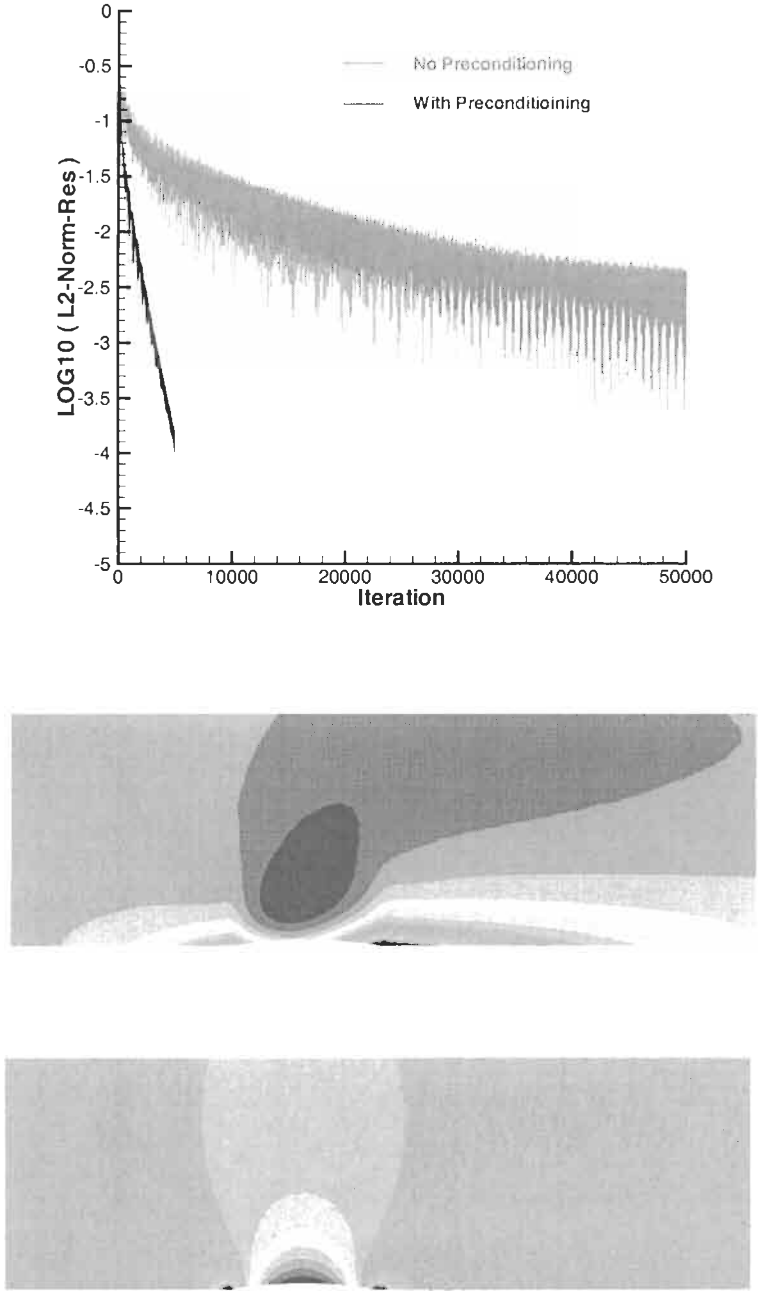


Figure 9.7: Inviscid 3-D flow in a channel with bump. Unstructured hybrid grid (prisms and tetrahedra), $M_{inlet} = 10^{-3}$, 2nd-order Roe's upwind discretisation, explicit multistage time-stepping scheme. Shown are (top to bottom) the convergence history, the solution without and with preconditioning.

Bibliography

[1] Kroll, N.; Jain, R.K.: *Solution of Two-Dimensional Euler Equations - Experience with a Finite Volume Code.* DFVLR-FB 87-41, 1987.

[2] Mavriplis, D.J.: *On Convergence Acceleration Techniques for Unstructured Meshes.* AIAA Paper 98-2966, 1998.

[3] Jameson, A.: *Iterative Solution of Transonic Flow over Airfoil and Wings Including at Mach 1.* Comm. Pure Appl. Math., 27 (1974), pp. 283-309.

[4] Jameson, A.; Schmidt, W.; Turkel, E.: *Numerical Solutions of the Euler Equations by Finite Volume Methods Using Runge-Kutta Time-Stepping Schemes.* AIAA Paper 81-1259, 1981.

[5] Van Leer, B.; Tai, C.H.; Powell, K.G.: *Design of Optimally Smoothing Multi-Stage Schemes for the Euler Equations.* AIAA Paper 89-1933, 1989.

[6] Tai, C.-H.; Sheu, J.-H.; van Leer, B.: *Optimal Multistage Schemes for Euler Equations with Residual Smoothing.* AIAA Journal, 33 (1995), pp. 1008-1016.

[7] Tai, C.-H.; Sheu, J.-H.; Tzeng, P.-Y.: *Improvement of Explicit Multistage Schemes for Central Spatial Discretization.* AIAA Journal, 34 (1996), pp. 185-188.

[8] Jameson, A.; Baker, T.J.: *Solution of the Euler Equations for Complex Configurations.* AIAA Paper 83-1929, 1983.

[9] Enander, R.; Karlsson, A.R.: *Implicit Explicit Residual Smoothing in Multigrid Cycle.* AIAA Paper 95-0204, 1995.

[10] Enander, R.; Sjögreen, B.: *Implicit Explicit Residual Smoothing for Upwind Schemes.* Uppsala University, Report No. 179, 1996.

[11] Martinelli, L.: *Calculation of Viscous Flows with a Multigrid Method.* Ph.D. Thesis, Dept. of Mechanical and Aerospace Eng., Princeton University, 1987.

[12] Turkel, E.; Swanson, R.C.; Vatsa, V.N.; White J.A.: *Multigrid for Hypersonic Viscous Two- and Three-Dimensional Flows.* AIAA Paper 91-1572, 1991.

[13] Radespiel, R.; Kroll, N.: *Multigrid Schemes with Semicoarsening for Accurate Computations of Hypersonic Viscous Flows.* DLR Internal Report, No. 129-90/19, 1991.

[14] Jameson, A.; Baker, T.J.; Weatherill, N.P.: *Calculation of Inviscid Transonic Flow over a Complete Aircraft.* AIAA Paper 86-0103, 1986.

[15] Blazek, J.; Kroll, N.; Radespiel, R.; Rossow, C.-C.: *Upwind Implicit Residual Smoothing Method for Multi-Stage Schemes.* AIAA Paper 91-1533, 1991.

[16] Blazek, J.: *Methods to Accelerate the Solutions of the Euler- and Navier-Stokes Equations for Steady-State Super- and Hypersonic Flows.* Translation of DLR Research Report 94-35, ESA-TT-1331, 1995.

[17] Blazek, J.; Kroll, N.; Rossow, C.-C.: *A Comparison of Several Implicit Smoothing Methods.* ICFD Conf. Numerical Methods for Fluid Dynamics, Reading, England, 1992.

[18] Blazek, J.; Kroll, N.; Rossow, C.-C.; Swanson, R.C.: *A Comparison of Several Implicit Residual Smoothing Methods in Combination with Multigrid.* 13th Int. Conf. on Numerical Methods in Fluid Dynamics, Roma, Italy, 1992.

[19] Brandt, A.: *Guide to Multigrid Development. Multigrid Methods I* Lecture Notes in Mathematics, No. 960, Springer Verlag, New York, 1981.

[20] Jameson, A.: *Solution of the Euler Equations for Two-Dimensional Transonic Flow by a Multigrid Method.* MAE Report No. 1613, Dept. of Mechanical and Aerospace Engineering, Princeton University, 1983.

[21] Jameson, A.: *Solution of the Euler Equations by a Multigrid Method.* Applied Mathematics and Computation, 13 (1983), pp. 327-356.

[22] Jameson, A.: *Multigrid Algorithms for Compressible Flow Calculations.* Multigrid Methods II., Lecture Notes in Mathematics, No. 1228, Springer Verlag, 1985.

[23] Schröder, W.; Hänel, D.: *An Unfactored Implicit Scheme with Multigrid Acceleration for the Solution of the Navier-Stokes Equations.* J. Computers and Fluids, 15 (1987), pp. 313-320.

[24] Koren, B.: *Multigrid and Defect Correction for the Steady Navier-Stokes Equations.* J. Computational Physics, 87 (1990), pp. 25-46.

[25] Thomas, J.L.: *An Implicit Multigrid Scheme for Hypersonic Strong-Interaction Flowfields.* 5th Copper Mountain Conf. on Multigrid Methods, Denver, USA, 1991.

[26] Radespiel, R.; Swanson, R.C.: *Progress with Multigrid Schemes for Hypersonic Flow Problems.* ICASE Report, No. 91-89, 1991; also J. Computational Physics, 116 (1995), pp. 103-122.

[27] Mavriplis, D.J.; Venkatakrishnan, V.: *A Unified Multigrid Solver for the Navier-Stokes Equations on Mixed Element Meshes.* ICASE Report No. 95-53, 1995.

[28] Mavriplis, D.J.; Venkatakrishnan, V.: *A 3D Agglomeration Multigrid Solver for the Reynolds-Averaged Navier-Stokes Equations on Unstructured Meshes.* Int. Journal for Numerical Methods in Fluids, 23 (1996), pp. 527-544.

[29] Braaten, M.E.; Connell, S.D.: *Three-Dimensional Unstructured Adaptive Multigrid Scheme for the Navier-Stokes Equations.* AIAA Journal, 34 (1996), pp. 281-290.

[30] Mavriplis, D.J.: *Multigrid Strategies for Viscous Flow Solvers on Anisotropic Unstructured Meshes.* J. Computational Physics, 145 (1998), pp. 141-165.

[31] Haselbacher, A.C.: *A Grid-Transparent Numerical Method for Compressible Viscous Flows on Mixed Unstructured Grids.* PhD Thesis, Loughborough University, England, 1999.

[32] Yoon, S.; Jameson, A.: *A Multigrid LU-SSOR Scheme for Approximate Newton-Iteration Applied to the Euler Equations.* NASA-CR-17954, 1986.

[33] Yoon, S.; Jameson, A.: *Lower-Upper Implicit Schemes with Multiple Grids for the Euler Equations.* AIAA Paper 86-0105, 1986; also AIAA Journal, 7 (1987), pp. 929-935.

[34] Yoon, S.; Kwak, D.: *Multigrid Convergence of an Implicit Symmetric Relaxation Scheme.* AIAA Paper 93-3357, 1993.

[35] Blazek, J.: *Investigations of the Implicit LU-SSOR Scheme.* DLR Research Report, No. 93-51, 1993.

[36] Blazek, J.: *A Multigrid LU-SSOR Scheme for the Solution of Hypersonic Flow Problems.* AIAA Paper 94-0062, 1994.

[37] Roberts, T.W.; Sidilkover, D.; Swanson, R.C.: *Textbook Multigrid Efficiency for the Steady Euler Equations.* AIAA Paper 97-1949, 1997.

[38] Thomas, J.L.; Bonhaus, D.L.; Anderson, W.K.; Rumsey, C.L.; Biedron, R.T.: *An* $O(Nm^2)$ *Plane Solver for the Compressible Navier-Stokes Equations.* AIAA Paper 99-0785, 1999.

[39] Ruge, J.W.; Stüben, K.: *Algebraic Multigrid.* Multigrid Methods, SIAM Frontiers in Applied Math., McCormick S.F. (ed.), 1987, pp. 73-131.

[40] Lonsdale, R.D.: *An Algebraic Multigrid Solver for the Navier-Stokes Equations on Unstructured Meshes.* Int. J. Num. Meth. Heat Fluid Flow, 3 (1993), pp. 3-14.

[41] Webster, R.: *An Algebraic Multigrid Solver for Navier-Stokes Problems.* Int. J. Num. Meth. Heat Fluid Flow, 18 (1994), pp. 761-780.

[42] Raw, M.J.: *Robustness of Coupled Algebraic Multigrid for the Navier-Stokes Equations.* AIAA Paper 96-0297, 1996.

[43] Weiss, J.M.; Maruszewski, J.P.; Smith, W.A.: *Implicit Solution of Preconditioned Navier-Stokes Equations Using Algebraic Multigrid.* AIAA Journal, 37 (1999), pp. 29-36.

[44] Mulder, W.A.: *A New Multigrid Approach to Convection Problems.* J. of Computational Physics, 83 (1989), pp. 303-323.

[45] Lynn, J.F.; van Leer, B.: *A Semi-Coarsened Multigrid Solver for the Euler and Navier-Stokes Equations with Local Preconditioning.* AIAA Paper 95-1667, 1995.

[46] Mavriplis, D.J.: *Directional Agglomeration Multigrid Techniques for High Reynolds Number Viscous Flow Solvers.* AIAA Paper 98-0612, 1998.

[47] Hackbush, W.: *Multigrid Methods and Applications.* Springer Verlag, Berlin, Germany, 1985.

[48] Mavriplis, D.J.: *Multigrid Techniques for Unstructured Meshes.* ICASE Report No. 95-27, 1995.

[49] Mavriplis, D.J.: *Unstructured Grid Technique.* Annu. Rev. Fluid. Mech., 29 (1997), pp. 473-514.

[50] Leclerq, M.P.; Stoufflet, B.: *Characteristic Multigrid Method Application to Solve the Euler Equations with Unstructured and Unnested Grids.* J. Computational Physics, 104 (1993), pp. 329-346.

[51] Mavriplis, D.J.: *Three-Dimensional Multigrid for the Euler Equations.* AIAA Journal, 30 (1992), pp. 1753-1761.

[52] Peraire, J.; Peiró, J.; Morgan, K.: *A 3D Finite-Element Multigrid Solver for the Euler Equations.* AIAA Paper 92-0449, 1992.

[53] Morano, E.; Dervieux, A.: *Looking for O(N) Navier-Stokes Solutions on Non-Structured Meshes.* Proc. 6th Copper Mountain Conf. on Multigrid Methods, 1993, pp. 449-464; also ICASE Report 93-26, 1993.

[54] Riemslagh, K.; Dick, E.: *A Multigrid Method for Steady Euler Equations on Unstructured Adaptive Grids.* Proc. 6th Copper Mountain Conf. on Multigrid Methods, 1993, pp. 527-542.

[55] Mavriplis, D.J.; Martinelli, L.: *Multigrid Solution of Compressible Turbulent Flow on Unstructured Meshes Using a Two-Equation Model.* Int. J. for Numerical Methods in Fluids, 18 (1994), pp. 887-914.

[56] Crumpton, P.I.; Giles, M.B.: *Aircraft Computations Using Multigrid and an Unstructured Parallel Library.* AIAA Paper 95-0210, 1995.

[57] Guillard, H.: *Node-Nested Multigrid with Delaunay Coarsening.* INRIA Report No. 1898, 1993.

[58] Ollivier-Gooch, C.F.: *Multigrid Acceleration of an Upwind Euler Solver on Unstructured Meshes.* AIAA Journal, 33 (1995), pp. 1822-1827.

[59] Parthasarathy, V.; Kallinderis, Y.: *New Multigrid Approach for Three-Dimensional Unstructured, Adaptive Grids.* AIAA Journal, 32 (1994), pp. 956-963.

[60] Barth, T.J.: *Randomized Multigrid.* AIAA Paper 95-0207, 1995.

[61] Lawson, C.L.: *Software for C^1 Surface Interpolation.* Mathematical Software III, Rice, J.R. (ed.), Academic Press, 1977.

[62] Lawson, C.L.: *Properties of n-Dimensional Triangulations.* Computer Aided Geometric Design, 3 (1986), pp. 231-246.

[63] Crumpton, P.I.; Giles, M.B.: *Implicit Time Accurate Solutions on Unstructured Dynamic Grids.* AIAA Paper 95-1671, 1995.

[64] Moinier, P.; Müller J.-D.; Giles, M.B.: *Edge-Based Multigrid for Hybrid Grids.* AIAA Paper 99-3339, 1999.

[65] Lallemand, M.H.: *Schémas décentrés multigrilles pour la résolution des équations d'Euler en éléments finis.* PhD Thesis, Université de Provence, France, 1988.

[66] Lallemand, M.H.; Steve, H.; Dervieux, A.: *Unstructured Multigridding by Volume Agglomeration: Current Status.* Computers and Fluids, 21 (1992), pp. 397-433.

[67] Koobus, B.; Lallemand, M.H.; Dervieux, A.: *Unstructured Volume-Agglomeration Multigrid: Solution of the Poisson Equation.* Int. Journal for Numerical Methods in Fluids, 18 (1994), pp. 27-42.

[68] Smith, W.A.: *Multigrid Solution of Transonic Flow on Unstructured Grids.* Recent Advances and Applications in CFD, Proc. ASME Winter Annual Meeting, Dec. 1990, pp. 145-152.

[69] Venkatakrishnan, V.; Mavriplis, D.J.: *Agglomeration Multigrid for the Three-Dimensional Euler Equations.* AIAA Journal, 33 (1995), pp. 633-640.

[70] Mavriplis, D.J.; Venkatakrishnan, V.: *Agglomeration Multigrid for Viscous Turbulent Flows.* ICASE Report No. 94-62, 1994; also AIAA Paper 94-2332, 1994.

[71] Elias, S.R.; Stubley, G.D.; Raithby, G.D.: *An Adaptive Agglomeration Method for Additive Correction Multigrid.* Int. Journal for Numerical Methods in Engineering, 40 (1997), pp. 887-903.

[72] Okamoto, N.; Nakahashi, K.; Obayashi, S.: *A Coarse Grid Generation Algorithm for Agglomeration Multigrid Method on Unstructured Grids.* AIAA Paper 98-0615, 1998.

[73] Volpe, G.: *Performance of Compressible Flow Codes at Low Mach number.* AIAA Journal, 31 (1993), pp. 49-56.

[74] Lee, W.T.: *Local Preconditioning of the Euler Equations.* PhD Thesis, University of Michigan, 1991.

[75] Lee, D.: *Local Preconditioning of the Euler and Navier-Stokes Equations.* PhD Thesis, University of Michigan, 1996.

[76] Guillard, H.; Viozat, C.: *On the Behavior of Upwind Schemes in the Low Mach Number Limit.* INRIA Report No. 3160, 1997.

[77] Patankar, S.V.: *Numerical Heat Transfer and Fluid Flow.* McGraw-Hill, New York, 1980.

[78] Chorin, A.J.: *A Numerical Method for Solving Incompressible Viscous Flow Problems.* J. Computational Physics, 2 (1967), pp. 12-26.

[79] Turkel, E.: *Preconditioned Methods for Solving the Incompressible and Low Speed Compressible Equations.* J. Computational Physics, 72 (1987), pp. 277-298.

[80] Rogers, S.E.; Kwak, D.: *Upwind Differencing Scheme for the Time-Accurate Incompressible Navier-Stokes Equations.* AIAA Journal, 28 (1990), pp. 253-262.

[81] Cabuk, H.; Sung, C.-H.; Modi, V.: *Explicit Runge-Kutta Method for Three-Dimensional Internal Incompressible Flows.* AIAA Journal, 30 (1992), pp. 2024-2031.

[82] Shi, J.; Toro, E.F.: *A Riemann-Problem-Based Approach for Steady Incompressible Flows.* Int. J. Num. Meth. Heat Fluid Flow, 6 (1996), pp. 81-93.

[83] Liu, C.; Zheng, X.; Sung, C.H.: *Preconditioned Multigrid Methods for Unsteady Incompressible Flows.* J. Computational Physics, 139 (1998), pp. 35-57.

[84] Van Leer, B.; Lee, W.T.; Roe, P.: *Characteristic Time-Stepping or Local Preconditioning of the Euler Equations.* AIAA Paper 91-1552, 1991.

[85] Choi, Y.H.; Merkle, C.L.: *Time-Derivative Preconditioning for Viscous Flows.* AIAA Paper 91-1652, 1991.

[86] Choi, Y.H.; Merkle, C.L.: *The Application of Preconditioning in Viscous Flows.* J. Computational Physics, 105 (1993), pp. 207-223.

[87] Turkel, E.: *A Review of Preconditioning Methods for Fluid Dynamics.* Appl. Numerical Mathematics, 12 (1993), pp. 257-284.

[88] Turkel, E.; Vatsa, V.N.; Radespiel, R.: *Preconditioning Methods for Low-Speed Flows.* AIAA Paper 96-2460, 1996.

[89] Turkel, E.: *Preconditioning Techniques in Computational Fluid Dynamics.* Annu. Rev. Fluid Mech., 31 (1999), pp. 385-416.

[90] Lee, D.; van Leer, B.: *Progress in Local Preconditioning of the Euler and Navier-Stokes Equations.* AIAA Paper 93-3328, 1993.

[91] Lee, D.; van Leer, B.; Lynn, J.F.: *A Local Navier-Stokes Preconditioner for All Mach and Cell Reynolds Numbers.* AIAA Paper 97-2024, 1997.

[92] Jorgenson, P.C.; Pletcher, R.H.: *An Implicit Numerical Scheme for the Simulation of Internal Viscous Flow on Unstructured Grids.* Computers and Fluids, 25 (1996), pp. 447-466.

[93] Weiss, J.; Smith, W.A.: *Preconditioning Applied to Variable and Constant Density Flows.* AIAA Journal, 33 (1995), pp. 2050-2057.

[94] Godfrey, A.G.; Walters, R.W.; van Leer, B.: *Preconditioning for the Navier-Stokes Equations with Finite-Rate Chemistry.* AIAA Paper 93-0535, 1993.

[95] Radespiel, R.; Turkel, E.; Kroll, N.: *Assessment of Preconditioning Methods.* DLR Research Report No. 95-29, 1995.

[96] Dailey, L.D.; Pletcher, R.H.: *Evaluation of Multigrid Acceleration for Preconditioned Time-Accurate Navier-Stokes Algorithms.* Computers and Fluids, 25 (1996), pp. 791-811.

[97] Jespersen, D.; Pulliam, T.; Buning, P.: *Recent Enhancements to OVERFLOW.* AIAA Paper 97-0644, 1997.

[98] Sharov, D.; Nakahashi, K.: *Low Speed Preconditioning and LU-SGS Scheme for 3-D Viscous Flow Computations on Unstructured Grids.* AIAA Paper 98-0614, 1998.

[99] Currie, T.C.: *Modified Flux-Difference Splitting for Simulating Low Mach Number Flows, Including Combustion, on Unstructured Grids.* AIAA Paper 98-0230, 1998.

Chapter 10

Consistency, Accuracy and Stability

Within the finite-volume methodology, the surface integral in the Navier-Stokes equations (2.19) is somehow approximated for each control volume. We considered various possible spatial discretisations in Chapters 4 and 5. This approximation of the fluxes across the boundaries of the control volume causes a certain spatial *discretisation error*. This means that the discretised equations differ from the exact equations by the discretisation error, which results from the numerical scheme applied. Therefore, the important question is whether and how fast the solution of the discretised equations converges to the exact solution of the governing equations with increasingly finer grid. We shall discuss this question in the next two sections on the consistency and accuracy.

The solution of the time-dependent governing equations (2.19) requires also the discretisation of the time derivative of the conservative variables. We described several temporal discretisation methods in Chapter 6. We mentioned that each of the schemes has a specific order of accuracy and that there are certain limitations on the maximum size of the time step. These limitations can be assessed by means of the von Neumann stability analysis, which will be presented in Section 10.3. In this section, we shall also investigate the ability of the explicit multistage time-stepping scheme and of a generic implicit scheme to damp solution errors. The damping properties decide about the robustness of a particular scheme. They are also important for the success or failure of the multigrid acceleration (Section 9.4).

10.1 Consistency Requirements

A discretisation scheme is called *consistent*, if the discretised equations converge to the given differential equations for both the time step and grid size tending to zero. A consistent scheme gives us the security that we really solve the governing equations and nothing else. This is quoted in Ref. [1] as "solving the equations right" and is called *verification*. Verification should not be confused with the term *validation*. Validation means: do we solve "the right equations"? Thus verification tries to quantify the **numerical** errors, whereas validation deals with the **modelling** errors.

The consistency of a numerical scheme can be checked by expanding the function values into Taylor series. The developments are then inserted back into the discretised equations. If we subtract the differential equations, we obtain terms which represent the numerical error – the so-called *truncation error*. For a consistent scheme, the truncation error should go to zero with decreasing time step and grid size.

We will now illustrate this concept on a simple example. Let us consider the following 1-D scalar equation

$$\frac{\partial U}{\partial t} + \frac{\partial U}{\partial x} = 0\,. \tag{10.1}$$

A very simple but valid discretisation scheme would be

$$\frac{U_i^{n+1} - U_i^n}{\Delta t} + \frac{U_i^n - U_{i-1}^n}{\Delta x} = 0\,, \tag{10.2}$$

where n denotes the time level and i the node index, respectively. Expanding the solution U_i^{n+1} around the time level n gives

$$U_i^{n+1} = U_i^n + \Delta t \left(\frac{\partial U}{\partial t}\right)_i^n + \frac{(\Delta t)^2}{2}\left(\frac{\partial^2 U}{\partial t^2}\right)_i^n + \cdots \tag{10.3}$$

where $(\cdots)$ represents the higher-order terms. The Taylor series for the solution U_{i-1}^n reads (cf. Eq. (3.1))

$$U_{i-1}^n = U_i^n - \Delta x \left(\frac{\partial U}{\partial x}\right)_i^n + \frac{(\Delta x)^2}{2}\left(\frac{\partial^2 U}{\partial x^2}\right)_i^n + \cdots \tag{10.4}$$

If we substitute Eq. (10.3) and (10.4) into the discretised equation (10.2), we obtain

$$\left(\frac{\partial U}{\partial t} + \frac{\partial U}{\partial x}\right)_i^n = -\frac{\Delta t}{2}\left(\frac{\partial^2 U}{\partial t^2}\right)_i^n + \frac{\Delta x}{2}\left(\frac{\partial^2 U}{\partial x^2}\right)_i^n + \cdots \tag{10.5}$$

A comparison with the differential equation (10.1) shows that the terms on the right-hand side of Eq. (10.5) represent the truncation error, which is of the order $\mathcal{O}(\Delta t, \Delta x)$. The numerical scheme (10.2) is consistent, since the truncation error tends to zero for $\Delta t \to 0$, $\Delta x \to 0$.

10.2 Accuracy of Discretisation

The accuracy of a discretisation scheme is connected to its truncation error. If, for instance, the leading term of the truncation error is proportional to Δx, we speak of *first-order* accurate spatial scheme. If the leading term behaves like $(\Delta x)^2$, the scheme is second-order accurate, and so on. Thus the example scheme Eq. (10.2) is 1st-order accurate in space and in time (Eq. (10.5)). This leads us to the condition that the numerical scheme must be at least 1st-order accurate in order to be consistent. Otherwise, the truncation error cannot be reduced by decreasing the values of Δt and Δx.

The question is now, how can we assess the truncation error in practice. Certainly, the expansion into the Taylor series is not adequate for complex numerical schemes with non-linear switches, limiters, etc. The following are the possible ways of error estimation [1]:

- additional solution(s) on different grid(s) – grid refinement or coarsening, unrelated grid(s);
- additional solution(s) on the same grid – higher- or lower-order accurate discretisation;
- solution of auxiliary PDE on the same grid;
- algebraic evaluations on the same grid.

The most common approach is to solve the governing equations on a series of grids with different cell sizes. If we happen to know the exact solution, we can easily quantify the error for each grid. The rate by which the truncation error decreases determines the accuracy of the discretisation. For example, if we halve the grid size in all coordinate directions and the error drops by a factor of four, the scheme is 2nd-order accurate. However, very often the exact physical solution is not known. In such a case, the order of accuracy can be estimated using the formula [2]

$$p = \frac{\ln\left(\dfrac{f_3 - f_2}{f_2 - f_1}\right)}{\ln(r)}, \tag{10.6}$$

where p denotes the accuracy, r the refinement (coarsening) ratio and f the numerical solution, respectively. Index 1 denotes the finest grid and index 3 the coarsest grid. If the coarsening ratio is not constant between the grids, a more general relation can be found in Ref. [1].

Besides the estimation of the truncation error, varying the grid resolution is also used to obtain what is called *grid converged solution*. This is achieved if the solution does not change (within a certain tolerance) with further grid refinement. In this respect, we speak of *grid convergence studies*. Although the grid convergence studies can be very time consuming, it is recommended always to check if the solution is grid converged.

10.3 Von Neumann Stability Analysis

Before a new discretisation method is implemented, it is important to know, at least approximately, how the method will influence the stability and the convergence behaviour of the numerical scheme. It was already frequently confirmed that the von Neumann method of stability analysis can deliver reliable assessment of the properties of a solution scheme. The methodology was developed at Los Alamos during the Second World War. However, it was briefly described first in 1947 by Cranck and Nicholson [3]. Later, it was also published in Ref. [4]. A very helpful introduction to the von Neumann stability analysis can be found in [5].

Von Neumann stability analysis is applicable to discretised **linear** partial differential equations under the assumption of periodic boundary conditions. It is based on the decomposition of the solution into a Fourier series. On the one hand, this allows the investigation of the stability of a solution scheme. On the other hand, the behaviour of the solution across the frequency spectrum can be examined in detail. Precisely this is of fundamental importance for the estimation of the convergence properties and robustness of a scheme, since the individual components (Fourier modes) of the solution error have to be reduced (damped) as quickly as possible. Therefore, we speak of the *damping properties* of a solution scheme.

Because the von Neumann analysis is limited to linear problems, the Euler or the Navier-Stokes equations have to be substituted with a suitable model equation. Two often employed 1-D model problems will be presented further below. The first one describes pure convection of a disturbance, which models the behaviour of the Euler equations. The second model equation contains additionally a diffusion term in order to simulate the behaviour of the Navier-Stokes equations.

10.3.1 Fourier Symbol and Amplification Factor

Before we proceed with the model problems, let us examine the von Neumann analysis for a general 1-D scalar linear equation. After the spatial discretisation of the fluxes, we obtain a system of ordinary differential equations in time (cf. Eq. (4.3))

$$\Delta t \left(\frac{d}{dt} U_i \right) = -\frac{\Delta t}{\Delta x} R_i \,, \tag{10.7}$$

where U_i denotes a scalar variable at point i, R_i stands for the residual, and Δx represents the grid size. Assuming periodic boundary conditions, we can expand the solution U in Eq. (10.7) into a finite Fourier series [5], [6]

$$U_i = \sum_{k=-N}^{N} \hat{U}_k \, e^{I p_k (i \Delta x)} \tag{10.8}$$

with the point index i running from 0 to N ($x_i = i\Delta x$), p_k being the wave number and I the imaginary unit. Due to the linearity of the model equation,

it is sufficient to consider only an individual (l-th) Fourier mode, i.e.,

$$\hat{U}\, e^{I p_l (i\Delta x)} \tag{10.9}$$

since the entire influence can be obtained by superposition. If we now insert the Fourier mode Eq. (10.9) into the discretised model equation (10.7), we get

$$\Delta t \frac{d}{dt}\left(\hat{U}\, e^{I p_l (i\Delta x)}\right) = -\frac{\Delta t}{\Delta x} R_i = -\frac{\Delta t}{\Delta x}\, z\, \hat{U} e^{I i \Phi} \tag{10.10}$$

with the definition of the *phase angle* $\Phi = p_l \Delta x$. The complex function z in Eq. (10.10) represents the so-called *Fourier symbol* of the spatial operator. Its form depends on the discretisation type (central, upwind) and on the order of accuracy.

An explicit or implicit time-stepping scheme used to solve Eq. (10.7) will be linearly stable, if the amplitude of any harmonic ($\hat{U}$) does not grow in time. This means that the amplitude of the new solution must be equal or smaller than the amplitude of the previous solution. Hence, the *amplification factor* is defined as

$$g = \frac{\hat{U}^{n+1}}{\hat{U}^n}\,. \tag{10.11}$$

If we introduce the *time-stepping operator* f, we can write the amplification factor in the general form

$$g = 1 - f z\,. \tag{10.12}$$

The time-stepping scheme will be linearly stable if $|g| \leq 1$. In cases, where the spatial and temporal discretisations are separated (method of lines), the domain of stability depends on the time-stepping scheme (f), not on the spatial discretisation (z). The Fourier symbol of the spatial operator z must lie within the domain of stability for all phase angles Φ. We speak of good damping properties if $|g|$ is well below unity. This means that the perturbations of the numerical solution are rapidly damped in time. Consequently, the time-stepping scheme converges faster to the steady-state solution than for $|g| \to 1$.

10.3.2 Convection Model Equation

In this case, the scalar linear model equation takes the form

$$\frac{\partial U}{\partial t} + \Lambda \frac{\partial U}{\partial x} = 0\,. \tag{10.13}$$

The convection velocity Λ (also the eigenvalue) is assumed to be constant and positive. In the following, we shall consider different spatial discretisation schemes applied to Eq. (10.7) and their corresponding Fourier symbols.

Central Scheme with Artificial Dissipation

According to Eqs. (4.48)-(4.50), the residual R_i in Eq. (10.7) takes the form

$$R_i = \frac{\Lambda}{2}(U_{i+1} - U_{i-1}) + \Lambda \epsilon^{(4)}(U_{i+2} - 4U_{i+1} + 6U_i - 4U_{i-1} + U_{i-2}) \tag{10.14}$$

with $\epsilon^{(4)}$ denoting the dissipation coefficient. In order to simplify the analysis, only the 4th-order differences were retained in Eq. (10.14). The Fourier symbol of the spatial operator is obtained according to Eq. (10.10) by inserting the harmonic Eq. (10.9) into Eq. (10.14)

$$z = \Lambda\left[I \sin\Phi + 4\epsilon^{(4)}(1 - \cos\Phi)^2\right]. \tag{10.15}$$

Upwind Scheme

Using the 1st-order upwind scheme (see Eq. (4.46)), the residual R_i becomes

$$R_i = \Lambda(U_i - U_{i-1}) \tag{10.16}$$

and the associated Fourier symbol reads

$$z = \Lambda\left[I \sin\Phi + (1 - \cos\Phi)\right]. \tag{10.17}$$

In the case of the 2nd-order upwind spatial discretisation, the residual in Eq. (10.7) is given by

$$R_i = \frac{\Lambda}{2}(3U_i - 4U_{i-1} + U_{i-2}). \tag{10.18}$$

The corresponding Fourier symbols z of the spatial operator appears as

$$z = \Lambda\left[I \sin\Phi\,(2 - \cos\Phi) + (1 - \cos\Phi)^2\right]. \tag{10.19}$$

10.3.3 Convection-Diffusion Model Equation

The combined convection-diffusion model equation can be written in the form

$$\frac{\partial U}{\partial t} + \Lambda\frac{\partial U}{\partial x} = \nu\,\frac{\partial^2 U}{\partial x^2}, \tag{10.20}$$

where ν represent the viscosity coefficient. The diffusion term $\partial^2 U/\partial x^2$ is normally approximated by 2nd-order central differences. Thus,

$$\nu\,\frac{\partial^2 U}{\partial x^2} \approx \frac{\Lambda_v}{\Delta x}(U_{i+1} - 2U_i + U_{i-1}) \tag{10.21}$$

with

$$\Lambda_v = \frac{\nu}{\Delta x} \tag{10.22}$$

being the viscous eigenvalue.

The Fourier symbol of the spatial operator is now composed of the convective and the diffusive part, i.e.,

$$z = z_c - z_v\,, \tag{10.23}$$

where

$$z_v = 2\Lambda_v(\cos\Phi - 1) \tag{10.24}$$

and z_c corresponds to one of the forms presented in Eq. (10.15), (10.17), or (10.19), respectively.

As we can conclude from Eq. (10.24), the diffusion term changes only the real part of the Fourier symbol. This is typical for any kind of diffusion or artificial dissipation (cf. Eq. (10.15)). Furthermore, we can see from Eq. (10.17) or (10.19) that the upwind discretisation of the convective term also causes the Fourier symbol to have a real part $(1 - \cos\Phi)$. Therefore, we speak of upwind dissipation. Only the central discretisation of the convective term adds no numerical dissipation, since the Fourier symbol is given by $(I \sin\Phi)$. However, the central scheme allows for the unwanted odd-even decoupling of the solution.

10.3.4 Explicit Time-Stepping

The application of an m-stage explicit time-stepping scheme to the discretised model problem Eq. (10.7) can be described in the following way (see Subsection 6.1.1)

$$\begin{aligned}
U_i^{(0)} &= U_i^n \\
U_i^{(k)} &= U_i^{(0)} - \alpha_k \frac{\Delta t}{\Delta x} R_i^{(k-1)}\,, \qquad k = 1, \cdots, m \\
U_i^{n+1} &= U_i^{(m)}\,,
\end{aligned} \tag{10.25}$$

where α_k denote the stage coefficients. If we substitute the variable U by its Fourier representation Eq. (10.8) and the residual by the Fourier symbol z, Equation (10.25) transforms to

$$\begin{aligned}
\hat{U}^{(0)} &= \hat{U}^n \\
\hat{U}^{(k)} &= \hat{U}^{(0)} - \alpha_k \frac{\Delta t}{\Delta x}\, z\, \hat{U}^{(k-1)}\,, \qquad k = 1, \cdots, m \\
\hat{U}^{n+1} &= \hat{U}^{(m)}\,.
\end{aligned} \tag{10.26}$$

The amplification factor of the above m-stage explicit scheme is given by Eq. (10.12). Based on Eq. (10.26) it can be shown that the Fourier symbol of the time-stepping operator f has the form [7]

$$\begin{aligned}
f = \frac{\Delta t}{\Delta x} \Bigg[& \alpha_m - \alpha_{m-1}\alpha_m \left(\frac{\Delta t}{\Delta x} z \right) \\
& + \cdots - (-1)^m \alpha_1 \alpha_2 \cdots \alpha_m \left(\frac{\Delta t}{\Delta x} z \right)^{m-1} \Bigg]\,,
\end{aligned} \tag{10.27}$$

provided the convective and the dissipative part of z are evaluated at each stage (so-called (m, m)-scheme). The derivation of f becomes more involved for the hybrid multistage schemes (Subsection 6.1.2). In the case of the (5,3)-scheme

(Eq. (6.7)), the Fourier symbol of the time-stepping operator reads [8]

$$f = \frac{\Delta t}{\Delta x}\{\alpha_5 - \alpha_5\alpha_4(z_I + \beta_5 z_R)(1 - \alpha_3 z_I)$$
$$-\alpha_5\alpha_4\alpha_2(1 - \alpha_1 z_I)(z_I + \beta_5 z_R)\left[\alpha_3(z_I + \beta_3 z_R)z_I - \beta_3 z_R\right] \tag{10.28}$$
$$-\alpha_5\alpha_2(1 - \beta_5)\beta_3(1 - \alpha_1 z_I)z_R\}$$

with

$$z_R = \frac{\Delta t}{\Delta x}\,\mathrm{Real}(z) \quad \text{and} \quad z_I = \frac{\Delta t}{\Delta x}\,\mathrm{Imag}(z) \tag{10.29}$$

representing the real and the imaginary part of z. The stage coefficients α_k and the blending coefficients β_k are shown in Table 6.2.

The time step Δt can be determined with the aid of the Courant-Friedrichs-Lewy (CFL) condition [9]. The following formula can be found for the convection model equation (10.13)

$$\Delta t = \sigma\,\frac{\Delta x}{|\Lambda|}\,. \tag{10.30}$$

The parameter σ in Eq. (10.30) denotes the CFL number. Its magnitude depends on the type and on the stage coefficients of the time-stepping scheme. The derivation of Eq. (10.30) is presented in Subsection 10.3.6. In this case of the convection-diffusion model equation (10.20), the relation

$$\Delta t = \sigma\,\frac{\Delta x}{|\Lambda| + C\Lambda_v} \tag{10.31}$$

holds for the time step. The factor C in Eq. (10.31) varies with the spatial discretisation. For the central scheme Eq. (10.15), $C = 4$ results in good damping. In the case of the 1st-order upwind scheme Eq. (10.17), $C = 2$ guarantees that the Fourier symbol of $z = z_c - z_v$ (Eq. (10.23)) remains bounded by z_c. In fact, the Fourier symbol with the diffusion term is identical to the Fourier symbol of z_c for $\Phi = \pi$, regardless of the ratio $\Lambda_v/|\Lambda|$. The same situation is encountered for the 2nd-order upwind scheme Eq. (10.19) if one sets $C = 1$.

Examples of Fourier Symbols and Amplification Factors

In the following, we shall consider a few applications of the von Neumann stability analysis to the convection and the mixed convection-diffusion model problems. The left-hand side of the figures below shows the locus of the Fourier symbol of the spatial operator z (thick line) together with the isolines of the magnitude of the amplification factor $|g|$. The boundary of the stability region is represented by $|g| = 1$. The behaviour of $|g|$ with respect to the phase angle Φ is displayed on the right-hand side. This allows the assessment of the damping properties.

Fourier symbols and damping of upwind discretisation schemes are displayed in Fig. 10.1 for the convection model equation. In both cases, a 3-stage scheme is employed with optimised coefficients (Table 6.1). The behaviour of the 1st-order

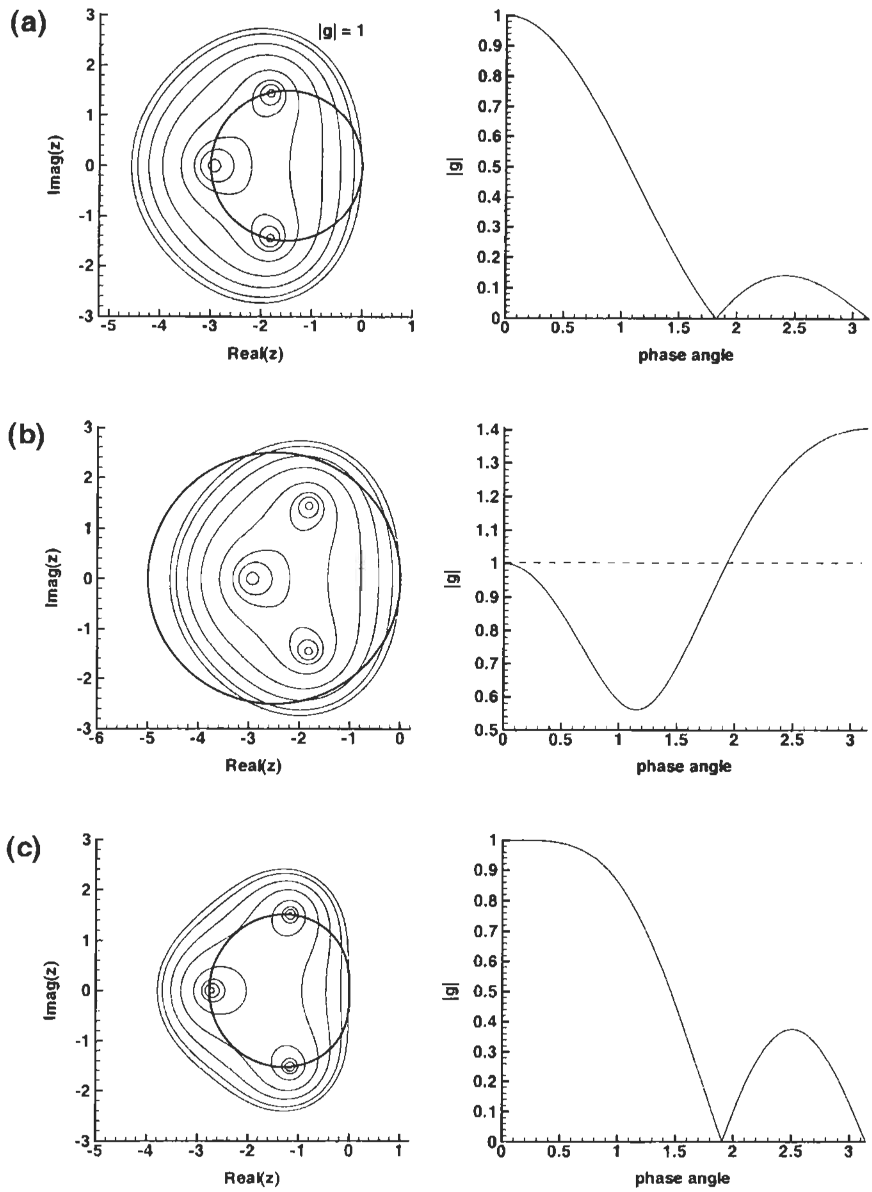


Figure 10.1: Convection model equation – Fourier symbol of the spatial operator (z) and the magnitude of the amplification factor ($|g|$) in the case of the explicit (3,3)-scheme:

(a) 1st-order upwind discretisation; $\sigma = 1.5$; stage coefficients: 0.1481, 0.4, 1.0
(b) like above but $\sigma = 2.5$
(c) 2nd-order upwind; $\sigma = 0.69$; stage coefficients: 0.1918, 0.4929, 1.0.

upwind scheme Eq. (10.16) is shown in Fig. 10.1a and 10.1b. As we can see, the locus of the Fourier symbol passes all three minima of $|g|$ (small circular areas on the isoplot). This leads to very low magnitude of the amplification factor for a phase angle $\Phi \geq \pi/2$, which is particularly important for an efficient multigrid scheme ($\Phi \geq \pi/2$ on coarse grid corresponds to $\Phi < \pi/2$ on fine grid). We can also observe that the damping properties are poor for $\Phi < \pi/2$. This behaviour is typical for the multistage schemes. It explains their low asymptotic convergence rate, which can be best improved by multigrid. It should be noted that the range $0 \leq \Phi \leq \pi$ represents the first half of the locus on the left-hand side.

The next diagram (Fig. 10.1b), demonstrates what happens, if the CFL-number σ is increased too much. As expected, the Fourier symbol extends behind the stability boundary. Consequently, the magnitude of g becomes larger than unity (dashed line). This means that the solution errors are amplified for the corresponding phase angles (frequencies). Such a time-stepping scheme will clearly diverge. Part (c) of Fig. 10.1 shows the properties of the 2nd-order upwind scheme Eq. (10.18). In principle, the behaviour is similar to that of the 1st-order upwind scheme.

The following plots in Fig. 10.2 display the loci of the Fourier symbols and the damping properties of the hybrid (5,3)-scheme (Subsection 6.1.2) applied to the convection model equation (10.13). The spatial discretisation utilises the central scheme with artificial dissipation (Eq. (10.14)). In order to demonstrate the influence of the dissipation coefficient $\epsilon^{(4)}$, its value is varied from 1/16 (Fig. 10.2a) to 1/256 (Fig. 10.2c). We can conclude from the results that the locus is contracted along the real axis with decreasing amount of artificial dissipation. This effect is caused by down-sizing the term $(1 - \cos\Phi)^2$ in Eq. (10.15). More important is the fact that the damping properties deteriorate with reduced artificial dissipation. This means in practice that the convergence speed and the robustness of the scheme will degrade with lower dissipation level.

The properties of the hybrid (5,3)-scheme employed to solve the convection-diffusion equation (10.20) are investigated next. At first, the scheme is coupled to the 1st-order upwind spatial discretisation Eq. (10.16). In Fig. 10.3, comparison is made between $\Lambda_v = 0$ (pure convection) and $\Lambda_v/\Lambda = 2$ (denoted by dashed line). As we can see, the locus of the Fourier symbol is contracted along the imaginary axis due to the influence of the diffusion term z_v, which has only a real component. The extension of the Fourier symbol along the real axis is kept, if the time step is calculated according to Eq. (10.31) with $C = 2$. The damping of the scheme remains on about the same favourable level as for pure convection. This is caused by the optimised, flat minimum of $|g|$ in the region surrounded by the locus of z_c (right-hand side of Fig. 10.3).

The behaviour of the hybrid (5,3)-scheme with central discretisation (Eq. (10.14)) is displayed in Fig. 10.4. The dissipation coefficient was set to $\epsilon^{(4)} = 1/64$ and the ratio of the eigenvalues to $\Lambda_v/\Lambda = 2$, respectively. We can observe that the locus of the Fourier symbol changes its form completely as compared to the convection equation. In the limit case $\Lambda_v \to \infty$, the locus would degrade to a line. Therefore, it is important that the time-stepping is optimised to have

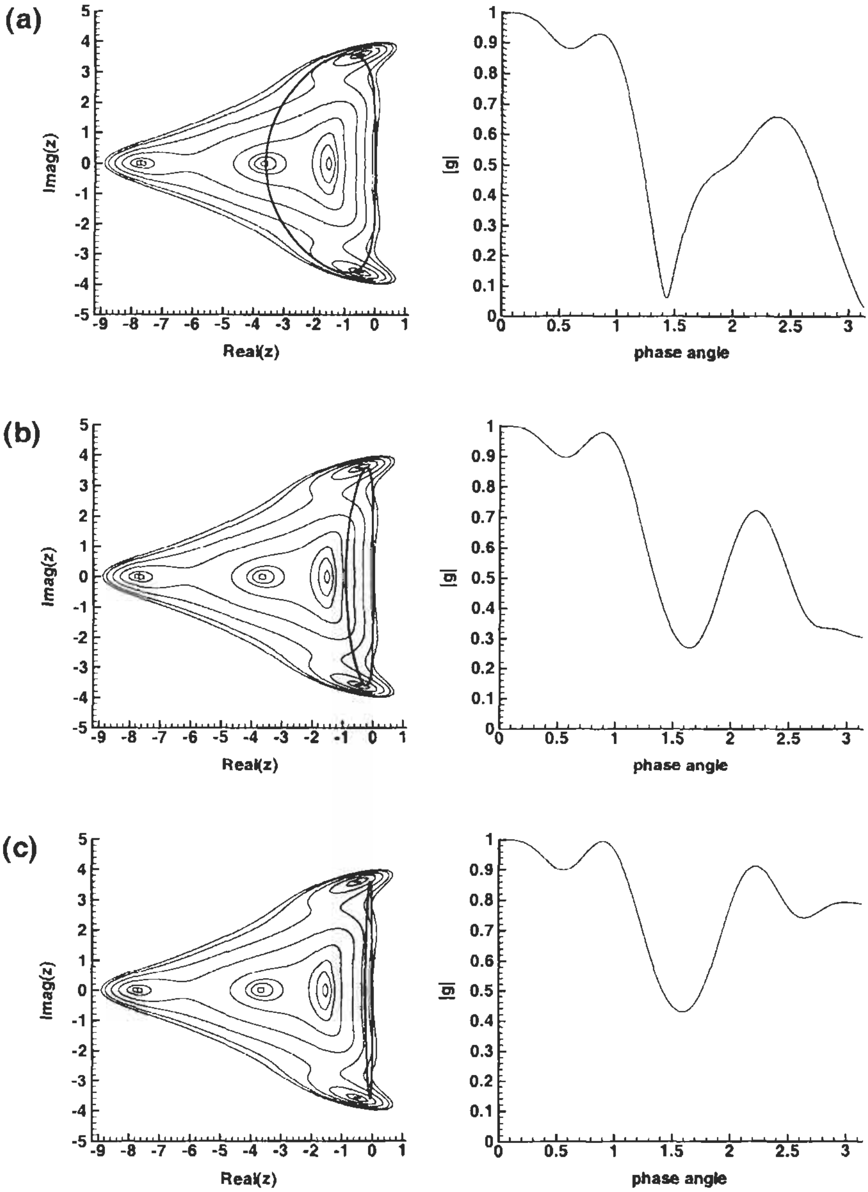


Figure 10.2: Convection model equation – Fourier symbol of the spatial operator (z) and the magnitude of the amplification factor ($|g|$) in the case of the explicit (5,3)-scheme with central spatial discretisation, $\sigma = 3.6$, stage and blending coefficients from Table 6.2:

(a) $\epsilon^{(4)} = 1/16$
(b) $\epsilon^{(4)} = 1/64$
(c) $\epsilon^{(4)} = 1/256$.

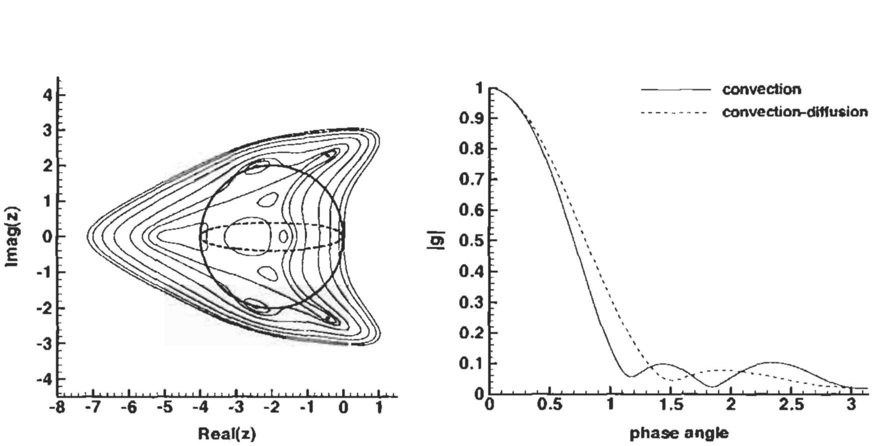


Figure 10.3: Convection-diffusion model equation – Fourier symbol of the spatial operator (z) and the magnitude of the amplification factor ($|g|$) in the case of the explicit (5,3)-scheme with 1st-order upwind spatial discretisation, $\sigma = 2.0$, stage and blending coefficients from Table 6.2.

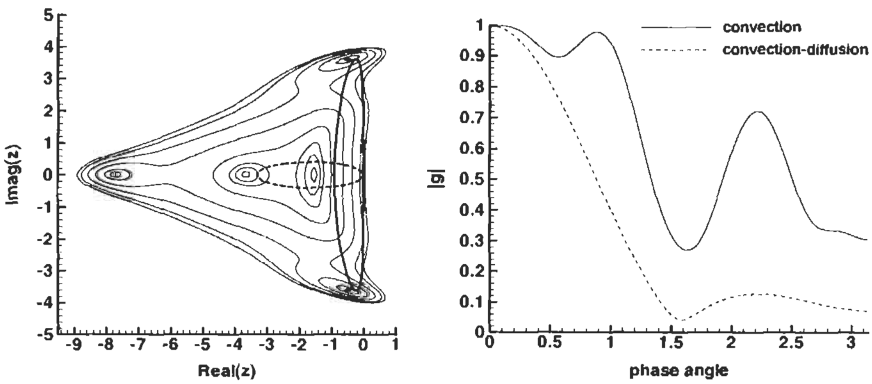


Figure 10.4: Convection-diffusion model equation – Fourier symbol of the spatial operator (z) and the magnitude of the amplification factor ($|g|$) in the case of the explicit (5,3)-scheme with central spatial discretisation, $\sigma = 3.6$, stage and blending coefficients from Table 6.2.

a shallow minimum of $|g|$ along the real axis. It should be mentioned that the time step was evaluated with Eq. (10.31) and $C = 4$.

Further examples related to the von Neumann stability analysis can be found in the Refs. [7], and [10]-[16]. Program for the linear analysis of explicit multistage schemes (with implicit residual smoothing) is provided on the CD-ROM.

10.3.5 Implicit Time-Stepping

According to Eq. (6.28) in Section 6.2, a general implicit scheme for the integration of the model equations can be formulated as

$$\left[\frac{\Delta x}{\Delta t} + \beta\left(\frac{\partial R}{\partial U}\right)_i\right]\Delta U^n = -R_i^n\,. \tag{10.32}$$

We can rewrite the flux Jacobian $\partial R/\partial U$ as a sum of two difference operators – one for the convection and one for the diffusion term in Eq. (10.20), i.e.,

$$\left\{\frac{\Delta x}{\Delta t} + \beta\left[(D_x^I)_c - (D_x^I)_v\right]_i\right\}\Delta U^n = -R_i^n\,. \tag{10.33}$$

The convection difference-operator can take various forms. For instance, in the case of the central scheme with artificial dissipation it becomes (cf. Eq. (6.47))

$$(D_x^I)_c\,\Delta U^n = \frac{\Lambda}{2}(\Delta U_{i+1}^n - \Delta U_{i-1}^n) + \Lambda\epsilon^I(\Delta U_{i+1}^n - 2\Delta U_i^n + \Delta U_{i-1}^n)\,, \tag{10.34}$$

where ϵ^I denotes the implicit dissipation coefficient. We do not include here the 4th-differences, since those are only seldom used in practice (high numerical effort). The 1st- or the 2nd-order upwind scheme in the implicit operator leads to

$$(D_x^I)_c\,\Delta U^n = \Lambda(\Delta U_i^n - \Delta U_{i-1}^n)\,, \tag{10.35}$$

or

$$(D_x^I)_c\,\Delta U^n = \frac{\Lambda}{2}(3\Delta U_i^n - 4\Delta U_{i-1}^n) + \Delta U_{i-2}^n)\,, \tag{10.36}$$

respectively. The diffusion difference-operator is usually of central type

$$(D_x^I)_v\,\Delta U^n = \frac{\Lambda_v}{2}(\Delta U_{i+1}^n - 2\Delta U_i^n + \Delta U_{i-1}^n)\,. \tag{10.37}$$

The discretisation of the explicit operator can be conducted accordingly to one of the relations (10.14), (10.16), or (10.18).

If we insert the Fourier mode Eq. (10.9) into the implicit scheme Eq. (10.33), we obtain

$$\left[\frac{\Delta x}{\Delta t} + \beta z^I\right]\Delta\hat{U}^n = -z^E\hat{U}^n\,, \tag{10.38}$$

where $z^I = z_c^I - z_v^I$ denotes the Fourier symbol of the flux Jacobian and $z^E = z_c^E - z_v^E$ represents the Fourier symbol of the explicit operator. The forms of

the Fourier symbols correspond to those derived in Sections 10.3.2 and 10.3.3. The amplification factor results with Eq. (10.12) as

$$g = 1 - \frac{z^E}{\left[\dfrac{\Delta x}{\Delta t} + \beta z^I\right]} . \tag{10.39}$$

Thus, the Fourier symbol of the time-stepping operator is $f = 1/[\cdots]$. A quick inspection of Eq. (10.39) reveals that for $\Delta t \to \infty$, $\beta = 1$ and $z^I = z^E$, the amplification factor will be zero for phase angles $\Phi > 0$. This situation occurs for the Newton scheme (see Subsection 6.2.5), if the flux Jacobian is exact. This explains the very fast convergence of the exact Newton's method.

Examples of Amplification Factors

In the following, we shall investigate the damping properties of a few implicit schemes with varying discretisations of the explicit and the implicit operator. For further discussion, the interested reader is referred to Ref. [17], which contains von Neumann analysis of the popular LU-SGS scheme in 2D (single grid and multigrid). A program for the analysis of implicit schemes is also provided on the accompanying CD-ROM.

In the first example, we consider a scheme which applies central spatial discretisation to the explicit (Eq. (10.14)) and the implicit operator (Eq. (10.34)). This is similar to the standard ADI scheme as presented in Subsection 6.2.3. We want to investigate the influence of parameter settings within the implicit operator on the amplification factor. Three exemplary results are compared in Fig. 10.5. The first curve (solid line) was generated with $\beta = 1$ and $\epsilon^I = 1/20$. The value of ϵ^I was chosen such that $|g| = 0$ at $\Phi = \pi$. The following formulae can be used to calculate ϵ^I

$$\epsilon^I = \frac{1}{\beta}\left(4\epsilon^{(4)} - \frac{\Delta x}{4|\Lambda|\Delta t}\right) . \tag{10.40}$$

The second curve (dashed line) demonstrates that the artificial dissipation has to be included in the implicit operator. For $\epsilon^I = 0$, the scheme becomes unstable at high frequencies ($\Phi \to \pi$). The last curve (dash-dotted) shows the magnitude of g for $\beta = 1/2$, i.e., for second-order accuracy in time. As we can see, the damping properties are much worser than for $\beta = 1$. Therefore, this value should be preferred. Unsteady flows are more efficiently solved by dual-time stepping (Section 6.3.2).

The effect of increasing CFL number is illustrated in Fig. 10.6. First-order upwind scheme (Eqs. (10.16) and (10.35)) is employed on both sides. The damping is very good due to the similarity between the explicit and the implicit operator. This confirms our remark with respect to Newton's scheme.

In the next diagram (Fig. 10.7), we compare the amplification factors for two different discretisations of the implicit operator (the left-hand side – LHS). The explicit operator is in both cases discretised using 2nd-order upwind scheme

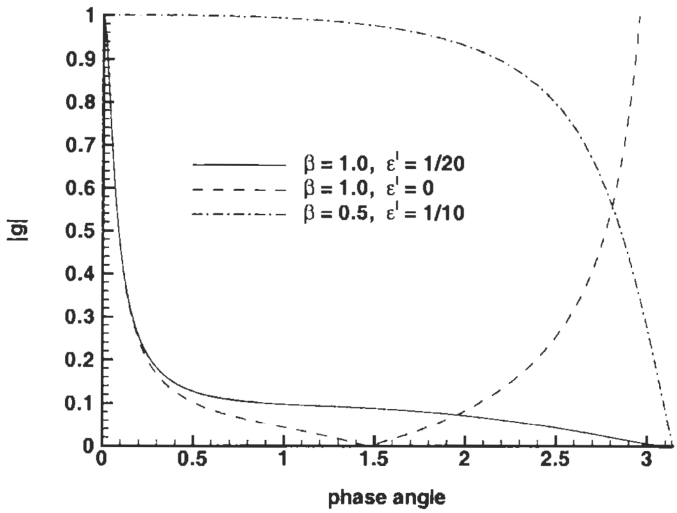


Figure 10.5: Convection model equation – magnitude of the amplification factor ($|g|$) in the case of an implicit scheme. Explicit and implicit operators discretised using central scheme, $\sigma = 20$, $\epsilon^{(4)} = 1/64$.

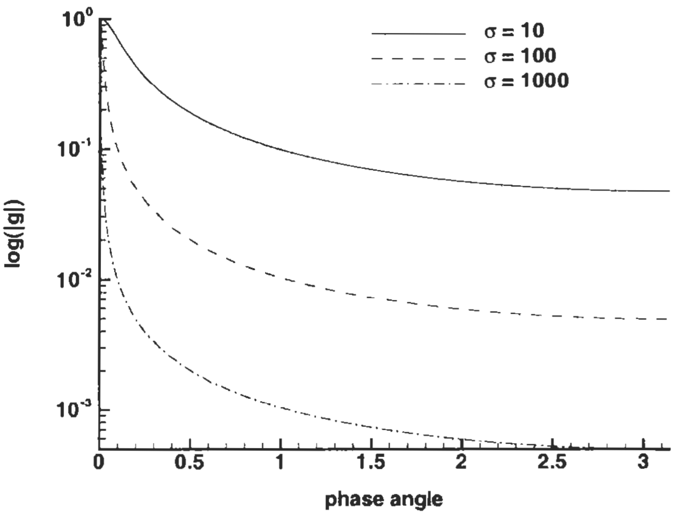


Figure 10.6: Convection model equation – magnitude of the amplification factor ($|g|$) in the case of an implicit scheme. Explicit and implicit operators discretised using 1st-order upwind scheme, $\beta = 1.0$. Note the logarithmic scaling of the y-axis.

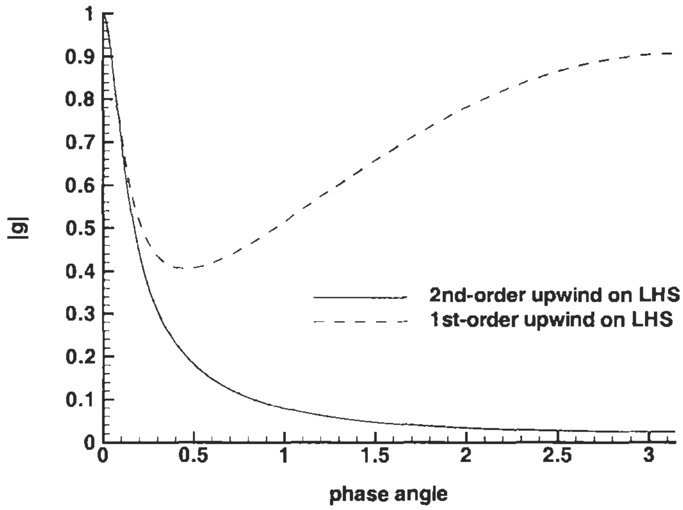


Figure 10.7: Convection model equation – magnitude of the amplification factor ($|g|$) in the case of an implicit scheme. Explicit operator discretised using 2nd-order upwind scheme, $\sigma = 10$, $\beta = 1.0$.

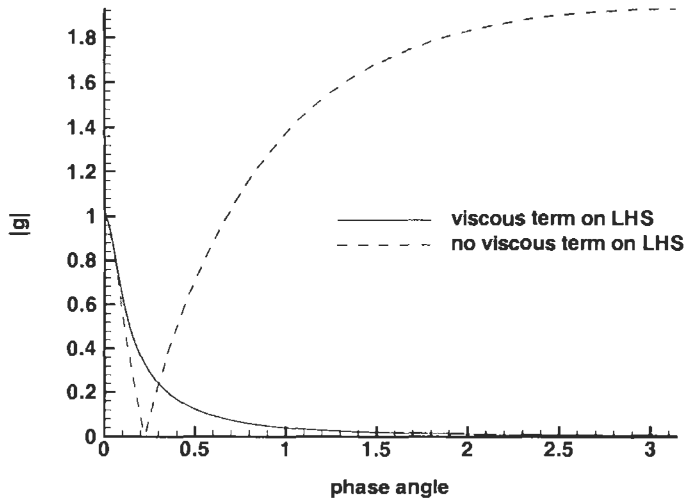


Figure 10.8: Convection-diffusion model equation – magnitude of the amplification factor ($|g|$) in the case of an implicit scheme. Explicit and implicit operators discretised using 2nd-order upwind scheme, $\sigma = 10$, $\beta = 1.0$, $\Lambda_v/\Lambda = 2$.

Eq. (10.18). It is evident that the exact representation of the explicit operator on the LHS leads to significantly better damping properties and thus to faster convergence. This observation was often confirmed in practice. The damping properties of a mixed 1st-/2nd-order discretisation can be improved by increased overrelaxation as demonstrated for the LU-SGS scheme [17], [18].

The last example in Fig. 10.8 deals with the convection-diffusion model equation (10.20). We want to compare two cases. In the first one, the discretisation of the diffusion term is also contained in the implicit operator (solid line). We can observe that the damping properties are very favourable – similar to those found for the convection model problem. However, if the diffusion term is removed from the implicit operator (dashed line in Fig. 10.8, the scheme becomes unstable, even at this low ratio of viscous to convective eigenvalue ($\Lambda_v/\Lambda = 2$). Therefore, the viscous fluxes should be always included in the approximation of the flux Jacobian to obtain a robust scheme.

In summary, we can state that the implicit operator should include at least the most important features of the spatial discretisation and of the physical problem. Rather crude approximations of the flux Jacobian $\partial\vec{R}/\partial\vec{W}$ can easily lead to an unstable scheme. Therefore, it is very important to find a reasonable compromise between the numerical effort and the accuracy of the numerical flux Jacobian.

A comparison to the results of the explicit multistage scheme reveals that the damping properties of a properly designed implicit scheme are significantly better. This is especially true for the damping at low phase angles (frequencies). Therefore, we can expect faster asymptotic convergence rates from an implicit scheme as compared to a multistage scheme.

10.3.6 Derivation of the CFL Condition

Every explicit time-stepping scheme remains stable only up to a certain value of the time step Δt. The necessary, but not sufficient, condition for stability of a time-stepping scheme was formulated by Courant, Friedrichs and Lewy [9]. The so-called CFL condition states that the domain of dependence of the numerical scheme has to include the domain of dependence of the partial differential equation. In order to clarify this statement, let us consider the x-t diagram in Fig. 10.9. The domain of dependence of the convection model equation (10.13) is given by the characteristic $dx/dt = \Lambda$. This means that any information is carried with this speed across the domain. Consequently, the exact solution at the time $(t_0 + \Delta t)$ is equal to the solution at the time t_0 but at the space coordinate $x^* = x_i - \Lambda\Delta t$. In order to simulate the behaviour of the exact solution correctly, the stencil of the spatial discretisation must enclose the point x^*. Thus, at least the point x_{i-1} has to be included (if $\Lambda > 0$). The domain of dependence of such a numerical scheme is represented by the shaded area in Fig. 10.9. Hence, we can formulate the condition

$$\Lambda\,\Delta t \leq \Delta x \quad \text{or} \quad \Delta t \leq \frac{\Delta x}{\Lambda}\,. \tag{10.41}$$

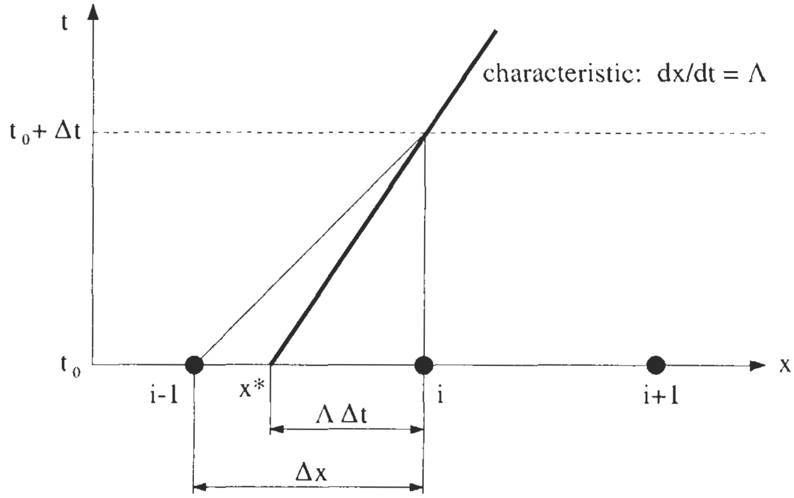


Figure 10.9: Domain of dependence of an explicit time-stepping scheme (shaded area) versus domain of dependence of the convective equation (thick line).

This leads us to the CFL condition

$$\sigma = |\Lambda| \frac{\Delta t}{\Delta x} \leq 1 \,. \tag{10.42}$$

Of course, the explicit multistage scheme allows for CFL-numbers $\sigma > 1$, since the new solution at the time $(t_0 + \Delta t)$ is obtained in more than one step.

Based on the above arguments, we can easily show that an implicit scheme like in Eq. (10.32) always fulfils the CFL condition, since the numerical domain of dependence extends over all grid points.

CFL Condition by von Neumann Analysis

The CFL condition in Eq. (10.42) can be also derived by the von Neumann stability analysis. Let us consider for this purpose the convection model equation (10.13) discretised using the 1st-order upwind scheme Eq. (10.16) and a one-stage explicit scheme. The domain of dependence of this numerical scheme corresponds to the shaded region in Fig. 10.9. According to Eq. (10.12), we obtain the amplification factor g from the relationship

$$g = 1 - \frac{\Delta t}{\Delta x} \, z \,, \tag{10.43}$$

since the Fourier symbol of the time-stepping operator reduces to $f = \Delta t / \Delta x$. If we substitute Eq. (10.17) for z in Eq. (10.43), the amplification factor will read

$$g = 1 - \frac{\Delta t}{\Delta x} \Lambda \big[I \sin \Phi + (1 - \cos \Phi) \big] \,. \tag{10.44}$$

Thus, the amplitude of g is given by (we assume $\Lambda > 0$)

$$|g|^2 = 2\frac{\Delta t}{\Delta x}\Lambda\left(\frac{\Delta t}{\Delta x}\Lambda - 1\right)(1 - \cos\Phi) + 1\,. \tag{10.45}$$

It is easy to show that the maximum of $|g|^2$ occurs at a phase angle $\Phi = \pi$. If we set $\Phi = \pi$ in Eq. (10.45), we obtain

$$|g(\Phi = \pi)|^2 = 4\frac{\Delta t}{\Delta x}\Lambda\left(\frac{\Delta t}{\Delta x}\Lambda - 1\right) + 1\,. \tag{10.46}$$

In order for the time-stepping scheme to be stable, it must hold that $|g|^2 \leq 1$. This condition can only be fulfilled if

$$\frac{\Delta t}{\Delta x}\Lambda \leq 1 \tag{10.47}$$

and hence

$$\Delta t \leq \frac{\Delta x}{\Lambda}\,. \tag{10.48}$$

This corresponds exactly to Eq. (10.41).

It is important to note that the CFL condition (10.42) is **not** sufficient (however necessary) to guarantee stability of the numerical scheme. Therefore, the von Neumann analysis should be carried out as well.

Bibliography

[1] Roache, P.J.: *Quantification of Uncertainty in Computational Fluid Dynamics.* Annu. Rev. Fluid Mech., 29 (1997), pp. 123-160.

[2] De Vahl, D.G.: *Natural Convection of Air in a Square Cavity: A Bench Mark Numerical Solution.* Int. J. Num. Meth. Fluids, 3 (1983), pp. 249-264.

[3] Cranck, J.; Nicholson, P.: *A Practical Method for Numerical Evaluation of Solutions of Partial Differential Equations of the Heat Conduction Type.* Proc. Cambridge Philosophical Soc., 43 (1947), pp. 50-67.

[4] Charney, J.G.; Fjortoft, R.; von Neumann, J.: *Numerical Integration of the Barotropic Vorticity Equation.* Tellus, 2 (1950), pp. 237-254.

[5] Hirsch, C.: *Numerical Computation of Internal and External Flows.* Vol. 1, John Wiley and Sons, 1988.

[6] Roache, P.J.: *Computational Fluid Dynamics.* Hermosa Publishers, Albuquerque, USA, 1972.

[7] Kroll, N.; Jain, R.K.: *Solution of Two-Dimensional Euler Equations – Experience with a Finite Volume Code.* DLR Research Report, No. 87-41, 1987.

[8] Radespiel, R.; Swanson, R.C.: *Progress with Multigrid Schemes for Hypersonic Flow Problems.* ICASE Report No. 91-89, 1991; also J. Computational Physics, 116 (1995), pp. 103-122.

[9] Courant, R.; Friedrichs, K.O.; Lewy, H.: *Über die partiellen Differenzengleichungen der mathematischen Physik.* Math. Ann., 100 (1928), pp. 32-74. Transl.: *On the Partial Difference Equations of Mathematical Physics.* IBM Journal, 11 (1967), pp. 215-234.

[10] Jameson, A.: *Multigrid Algorithms for Compressible Calculations.* Multigrid Methods II, Lecture Notes in Mathematics No. 1228, Springer Verlag, New York, 1985.

[11] Van Leer, B.; Tai, C.-H.; Powell, K.G.: *Design of Optimally Smoothing Multi-Stage Schemes for the Euler Equations.* AIAA Paper 89-1933, 1989.

[12] Lötstedt, P.; Gustafsson, B.: *Fourier Analysis of Multigrid Methods for General Systems of PDE.* Report No. 129/1990, Dept. Scientific Computing, Uppsala University, Sweden, 1990.

[13] Blazek, J.; Kroll, N.; Radespiel, R.; Rossow, C.-C.: *Upwind Implicit Residual Smoothing Method for Multi-Stage Schemes.* AIAA Paper 91-1533, 1991.

[14] Blazek, J.: *Methods to Accelerate the Solution of the Euler- and the Navier-Stokes Equations for Steady-State Super- and Hypersonic Flows.* Translation of DLR Research Report, No. 94-35, ESA-TT-1331, 1995.

[15] Tai, C.-H.; Sheu, J.-H.; van Leer, B.: *Optimal Multistage Schemes for Euler Equations with Residual Smoothing.* AIAA Journal, 33 (1995), pp. 1008-1016.

[16] Tai, C.-H.; Sheu, J.-H.; Tzeng, P.-Y.: *Improvement of Explicit Multistage Schemes for Central Spatial Discretization.* AIAA Journal, 34 (1996), pp. 185-188.

[17] Blazek, J.: *Investigations of the Implicit LU-SSOR Scheme.* DLR Research Report, No. 93-51, 1993.

[18] Blazek, J.: *A Multigrid LU-SSOR Scheme for the Solution of Hypersonic Flow Problems.* AIAA Paper 94-0062, 1994.

Chapter 11

Principles of Grid Generation

Prior to the numerical solution of the governing equations, we have to discretise the surfaces of all boundaries and to generate a volume grid inside the flow domain. As we discussed at the beginning of Chapter 3 (Section 3.1), we can choose basically between:

- *structured*, and
- *unstructured*

grids. An example of structured grid for a civil aircraft [1], [2] is presented in Fig. 11.1 and 11.2. For comparison, an unstructured surface and volume grid for a similar configuration [3]-[6] is displayed in Fig. 11.3 and 11.4.

The structured as well as the unstructured grids have their specific advantages and shortcomings, which we mentioned in Section 3.1. However, regardless of the grid type, the main bottleneck is currently the quality of data being imported from a CAD (Computer Aided Design) system into the grid generation program. The surface description is usually transfered via a standard format like IGES [7]. A direct transfer of CAD native data is rather rate [8]. The experience shows that this process can impair the accuracy of the data. Furthermore, the surface representation in the CAD system itself is often imprecise. This leads mostly to gaps, overlaps or discontinuities between neighbouring surface patches. Such errors have to be eliminated before the surfaces can be discretised. We speak in this respect of "CAD repair" [9], [10].

In the following, we shall present the basic methodologies applied to generate structured (Section 11.1) as well as unstructured (Section 11.2) grids. Due to the restricted space, we can provide here only a brief description. We refer the reader to [11] or [12] for a deeper discussion of surface modelling, structured and unstructured surface and volume grid generation.

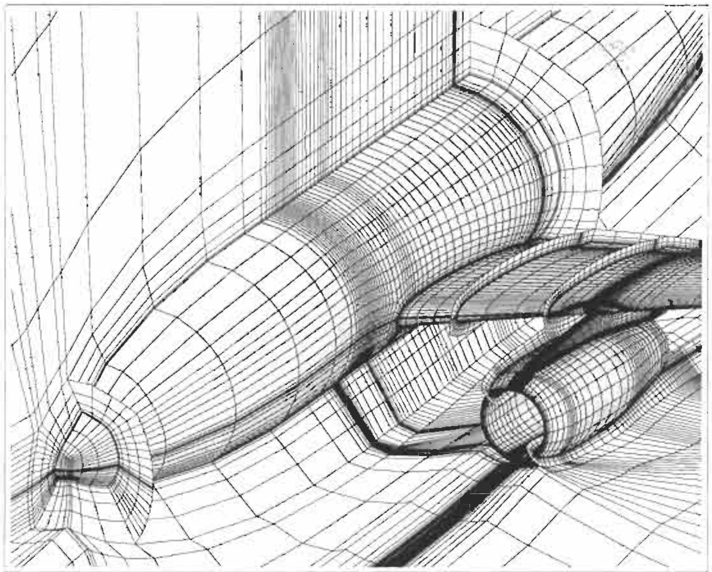


Figure 11.1: Structured surface and volume grid of a wing-body configuration. (Courtesy O. Brodersen, DLR, Germany).

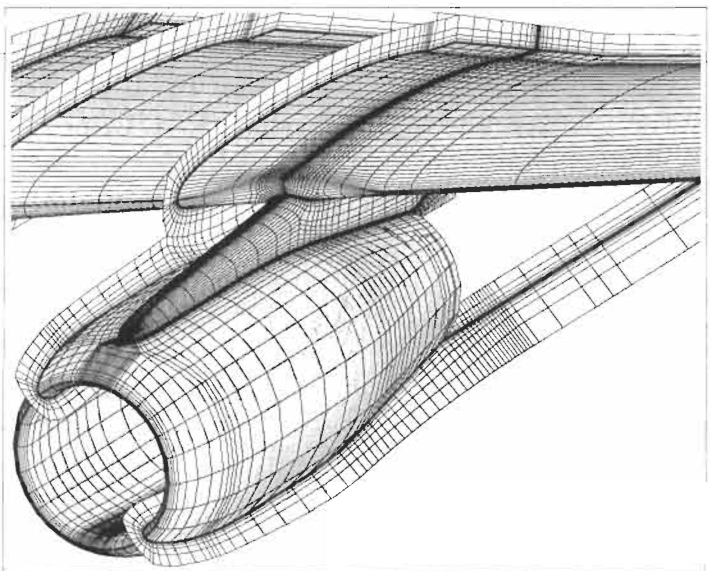


Figure 11.2: Structured surface and volume grid of a wing-body configuration – detail of the pylon and the engine nacelle. (Courtesy O. Brodersen, DLR, Germany).

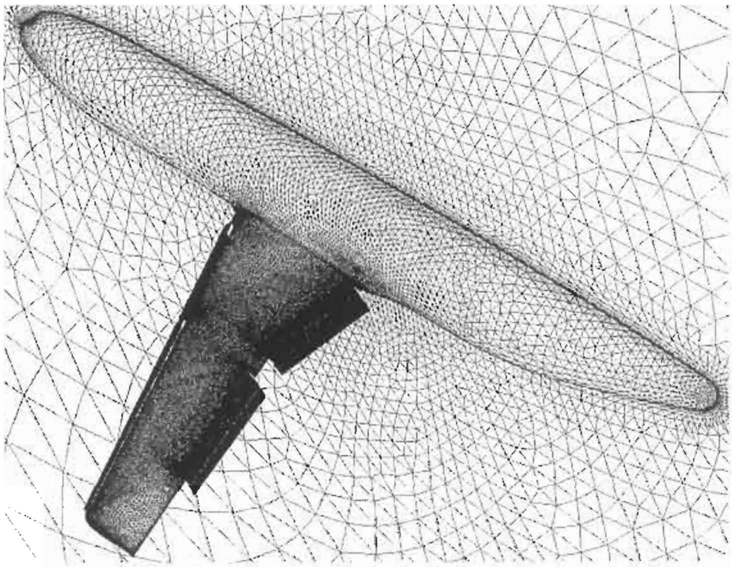


Figure 11.3: Unstructured surface grid of a wing-body configuration. (Courtesy D. Mavriplis, ICASE, USA).

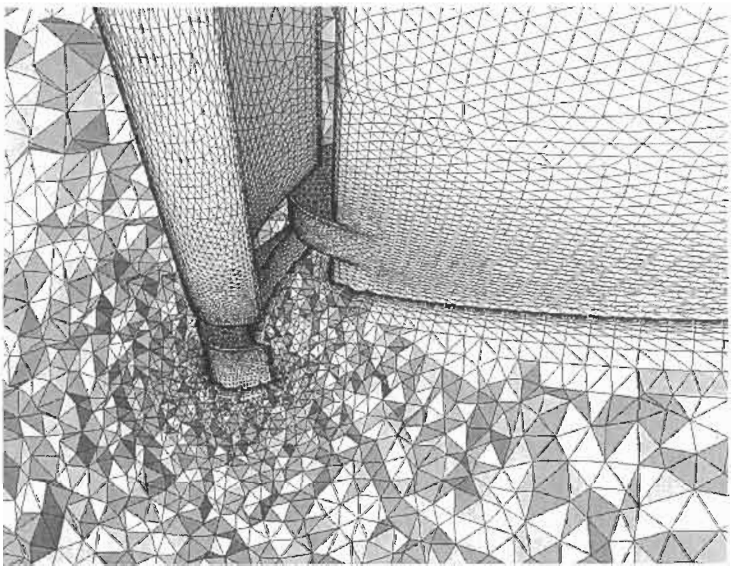


Figure 11.4: Unstructured grid of a wing-body configuration – detail of the slats and rough-cut through the volume grid. (Courtesy D. Mavriplis, ICASE, USA).

11.1 Structured Grids

The distinguishing feature of structured grids is that the grid points in the physical space are mapped in an unique way onto a continuous set of three integers i, j, k (one for each coordinate direction). The set of integers defines what is called the *computational* space (see Fig. 3.2). The coordinates ξ, η, ζ in the computational space are related to i, j, k as follows

$$\begin{aligned} \xi &= i/i_{max}\,, \quad & i &= 0,\,1,\,2,\,\cdots,\,i_{max} \\ \eta &= j/j_{max}\,, \quad & j &= 0,\,1,\,2,\,\cdots,\,j_{max} \\ \zeta &= k/k_{max}\,, \quad & k &= 0,\,1,\,2,\,\cdots,\,k_{max} \end{aligned} \tag{11.1}$$

This mapping implies that $0 \le \xi \le 1$, $0 \le \eta \le 1$, and $0 \le \zeta \le 1$. Neighbouring grid points can be connected to form cubes in the computational and hexahedra (quadrilaterals in 2D) in the physical space. Structured grid generation systems discretise the boundary surfaces of the flow domain using quadrilaterals – termed the *surface* grid – and fill the interior with hexahedra. The grid inside the domain is named the *volume* grid.

The generation of a structured grid starts by distributing grid points along boundary curves (boundaries of surface patches). The usual procedure is to place the nodes more dense in regions with high curvature. Using the point distribution on boundary curves, the surface grid can be generated. Based on the surface grids which enclose the physical domain, we can finally construct the volume grid. Thus, the common problem is to generate a grid inside the domain based on known boundary discretisation. This can be solved by two different approaches:

- *algebraic* grid generation, or
- grid generation using *partial differential* equations (PDE's).

The application of PDE's requires a valid initial grid (surface or volume), which is mostly generated algebraically. Two different types of PDE's are common:

- *elliptic* equations, and
- *hyperbolic* equations.

The algebraic grid generation (Subsection 11.1.2) employs a direct functional description of the coordinate transformation between the computational and the physical space. The most widely used algebraic technique is the so-called *Transfinite Interpolation* (TFI). Given the point distribution on all boundaries, it generates the grid points inside the physical domain by interpolation. Particular formulations of the TFI method allow for angle and grid spacing control at the boundaries.

The methodology based on elliptic PDE's (Subsection 11.1.3) is the most popular one. It allows the user to prescribe the angle between a grid line and boundary and to control the grid spacing and the expansion ratio near surfaces.

Elliptic grid generation also guarantees a smooth grid in the entire domain. Thus, high quality, boundary orthogonal grids can be generated. The downside of the elliptic method is a much longer computing time as compared to algebraic or hyperbolic grid generation approaches. The method also suffers from numerical difficulties.

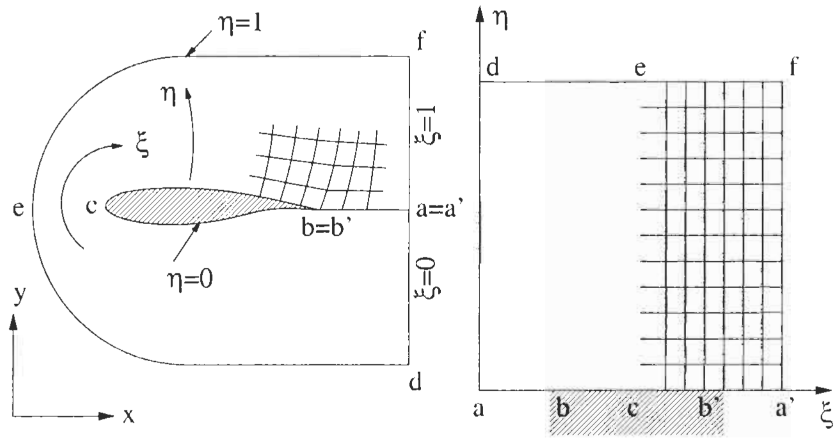
Hyperbolic PDE's can also be utilised for the grid generation (Subsection 11.1.4). This technique generates the volume grid by marching between two surfaces in the direction of one particular computational coordinate. The marching procedure is explicit, i.e., based on a known surface point distribution a new layer is generated. A natural restriction of the hyperbolic grid generation is that the shape of the outer grid boundary cannot be fully controlled. However, this may represent no real problem like in the case of external flows. The hyperbolic grid generation can provide approximate grid orthogonality over the entire domain. It is also computationally inexpensive.

11.1.1 C-, H-, and O-Grid Topology

Before we can start to generate any grid, we have to think about its topology. This means, we have to decide how many grid blocks are necessary and how the blocks should be ordered with respect to each other (by the way, this work may take weeks to months in the case of a complex geometry). For each grid block, we have to assign boundaries (or their parts) in the computational domain to particular boundaries in the physical space (e.g., solid wall, farfield, etc.). The appearance of the grid in the physical space will depend strongly on this assignment. In practice, three standard single-block grid topologies are established. They are named as the C-, H-, or the O-grid topology because in a plane view the grid lines resemble the corresponding capital letter. A grid in 3D can be described as a combination of two topologies. For example, the grid around a wing usually consists of a C-grid in the flow direction (cf. Fig. 11.2) and of an O-grid (or an H-grid) in the spanwise direction. In this case, we speak of a C-O-grid. In the following, we shall discuss all three topologies in more detail.

C-Grid Topology

In the case of the C-topology the aerodynamic body is enclosed by one family of grid lines, which also form the wake region (if present). The situation is sketched in Fig. 11.5. As we can see, the lines η = const. start at the farfield ($\xi = 0$), follow the wake, pass the trailing edge (node b), wrap in clockwise direction round the body, and finally continue to the farfield again ($\xi = 1$). The other family of grid lines (ξ = const.) emanates in normal direction from the body and the wake. The part (segment) a–b of the grid line $\eta = 0$ represents a *coordinate cut.* This means that the segment a–b in the physical space is mapped onto two segments in the computational space, namely $a \leq \xi \leq b$ and $b' \leq \xi \leq a'$. Hence, the nodes on the upper (b'–a') and the lower part (a–b) of the cut are hold separately in the computer memory. The appropriate boundary condition was presented in Section 8.6.



(a) physical space (b) computational space

Figure 11.5: C-grid topology in 2D.

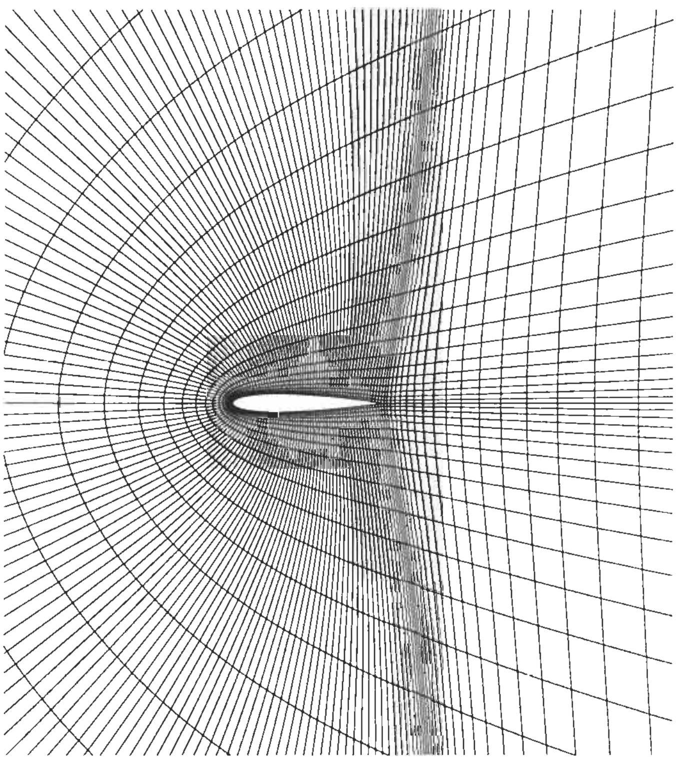


Figure 11.6: Partial view of a C-grid around NACA 0012 airfoil.

H-Grid Topology

The H-grid topology is quite often employed in turbomachinery for grid generation in the bladed flowpath. The topology is displayed in Fig. 11.7. As one can observe, the surface of the aerodynamic body is described here by two different grid lines, i.e., $\eta = 0$ and $\eta = 1$. On contrary to the C-grid, one family of grid lines ($\eta =$ const.) closely follows the streamlines (inlet located at $\xi = 0$, outlet at $\xi = 1$).

At the first sight, there is no obvious coordinate cut. However, in turbomachinery the segments a–b and e–f are periodic (rotationally periodic in 3D) to each other. The same is true for the segments c–d and g–h. This type of boundary condition is treated in Section 8.7. Figure 11.8 shows an example of a non-orthogonal H-grid between turbine blades.

O-Grid Topology

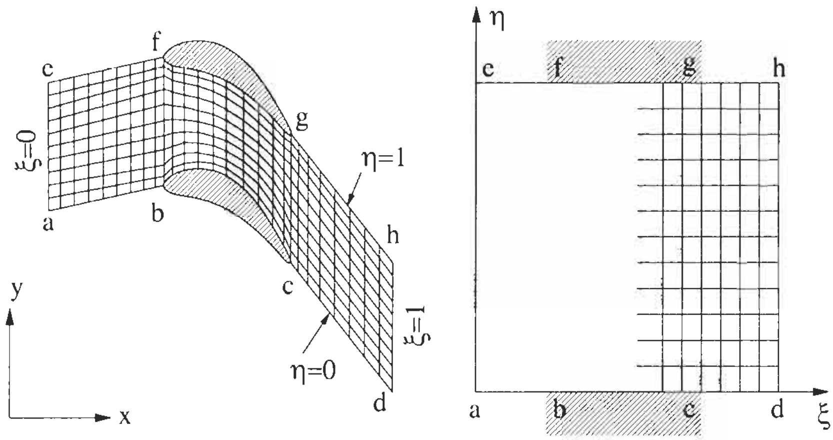
We can see from the rendering of the O-topology in Fig. 11.9 that one family of grid lines ($\eta =$ const.) forms closed curves around the aerodynamic body. The second family of grid lines ($\xi =$ const.) is spanned in radial direction between the body and the outer boundary (farfield in this case). The complete boundary line $\eta = 0$ represents the contour of the body (from a to a'). The coordinate cut is defined by the boundaries $\xi = 0$ (nodes a–c) and $\xi = 1$ (nodes a'–c') in the computational space. The example in Fig. 11.10 shows a standard O-grid used to simulate inviscid flow past an airfoil. A disadvantage of the O-topology is the poor grid quality at a sharp trailing edge.

11.1.2 Algebraic Grid Generation

The most widely used algebraic techniques for surface or volume grid generation from prescribed boundary point distribution is the Transfinite Interpolation (TFI) method. It was first described by Gordon and Hall in 1973 [13]. The TFI scheme utilises 1-D univariate interpolations in each of the coordinate directions in the computational space. The general form of the univariate interpolation functions reads

$$\begin{aligned}
\vec{U} &= \sum_{i=1}^{L}\sum_{n=0}^{P} \alpha_i^n(\xi)\,\frac{\partial^n \vec{r}(\xi_i, \eta, \zeta)}{\partial \xi^n} \\
\vec{V} &= \sum_{j=1}^{M}\sum_{m=0}^{Q} \beta_j^m(\eta)\,\frac{\partial^m \vec{r}(\xi, \eta_j, \zeta)}{\partial \eta^m} \\
\vec{W} &= \sum_{k=1}^{N}\sum_{l=0}^{R} \gamma_k^l(\zeta)\,\frac{\partial^l \vec{r}(\xi, \eta, \zeta_k)}{\partial \zeta^l}\,.
\end{aligned} \tag{11.2}$$

In Eq. (11.2), $\vec{U}$, $\vec{V}$, and $\vec{W}$ denote the univariate interpolation functions in the ξ-, η-, and ζ-direction, respectively. Furthermore, $\alpha_i^n(\xi)$, $\beta_j^m(\eta)$, $\gamma_k^l(\zeta)$ are the blending functions, and $\vec{r}$ stands for the position of a grid point in the



(a) physical space (b) computational space

Figure 11.7: H-grid topology in 2D.

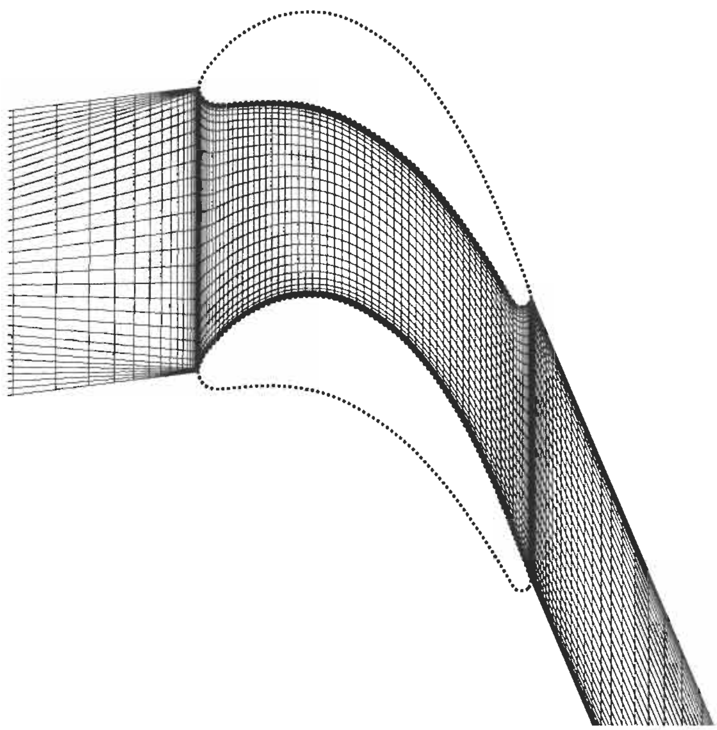
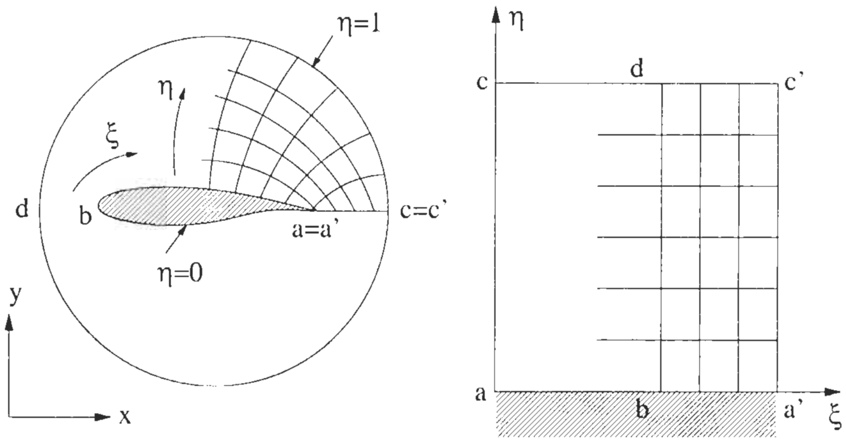


Figure 11.8: Partial view of an H-grid between turbine blades (dotted line).



(a) physical space (b) computational space

Figure 11.9: O-grid topology in 2D.

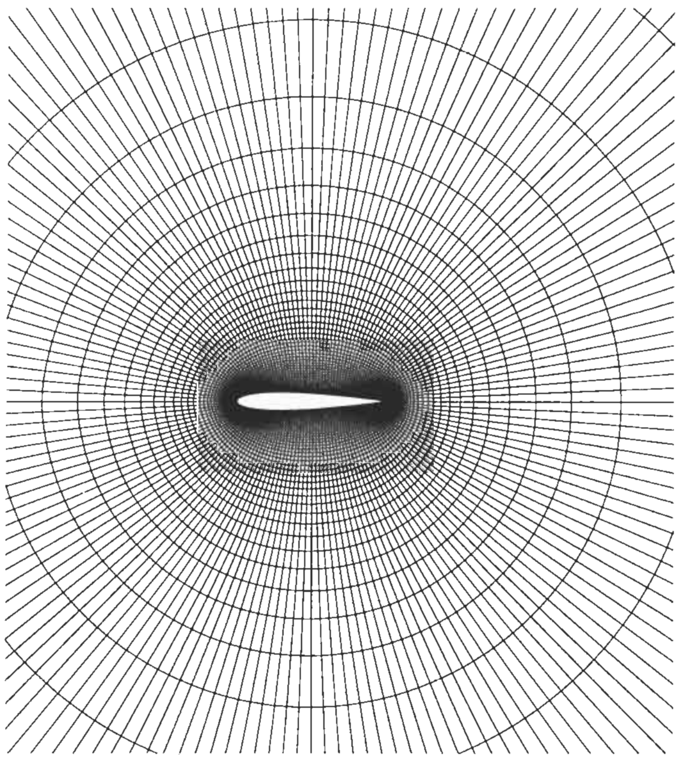


Figure 11.10: Partial view an O-grid around NACA 0012 airfoil.

physical space. In order to evaluate the interpolation functions, positions $\vec{r}$ and derivatives $\partial^n \vec{r}/\partial \xi^n$, etc. have to be specified. Since we already discretised the boundary curves or surfaces, we can insert these values into Eq. (11.2). With $\vec{U}$, $\vec{V}$, $\vec{W}$ known, we can generate the grid inside the domain by using the Boolean sum of the interpolation functions, i.e.,

$$\vec{r} = \vec{U} \oplus \vec{V} \oplus \vec{W} = \vec{U} + \vec{V} + \vec{W} - \vec{U}\vec{V} - \vec{U}\vec{W} - \vec{V}\vec{W} + \vec{U}\vec{V}\vec{W}\,. \tag{11.3}$$

The approach in Eq. (11.3) guarantees that in 2D all four boundary curves and in 3D all six boundary faces are matched. The tensor products in Eq. (11.3) are evaluated as follows

$$\begin{aligned}
\vec{U}\vec{V} &= \sum_{i=1}^{L}\sum_{j=1}^{M}\sum_{m=0}^{Q}\sum_{n=0}^{P} \alpha_i^n \beta_j^m \frac{\partial^{mn}\vec{r}(\xi_i, \eta_j, \zeta)}{\partial\eta^m \partial\xi^n} \\
\vec{U}\vec{W} &= \sum_{i=1}^{L}\sum_{k=1}^{N}\sum_{l=0}^{R}\sum_{n=0}^{P} \alpha_i^n \gamma_k^l \frac{\partial^{ln}\vec{r}(\xi_i, \eta, \zeta_k)}{\partial\zeta^l \partial\xi^n} \\
\vec{V}\vec{W} &= \sum_{j=1}^{M}\sum_{k=1}^{N}\sum_{l=0}^{R}\sum_{m=0}^{Q} \beta_j^m \gamma_k^l \frac{\partial^{lm}\vec{r}(\xi, \eta_j, \zeta_k)}{\partial\zeta^l \partial\eta^m} \\
\vec{U}\vec{V}\vec{W} &= \sum_{i=1}^{L}\sum_{j=1}^{M}\sum_{k=1}^{N}\sum_{l=0}^{R}\sum_{m=0}^{Q}\sum_{n=0}^{P} \alpha_i^n \beta_j^m \gamma_k^l \frac{\partial^{lmn}\vec{r}(\xi_i, \eta_j, \zeta_k)}{\partial\zeta^l \partial\eta^m \partial\xi^n}\,.
\end{aligned} \tag{11.4}$$

More details on Boolean operators and tensor products related to grid generation can be found in [14] or [15].

Various types of interpolation functions can be employed – linear, Lagrangian, Hermite, spline, etc. The most widespread method is the linear TFI. It is obtained by setting $L = M = N = 2$ and $P = Q = R = 0$ in Eqs. (11.2) and (11.4). With this, the volume grid can be generated based solely on the given point distribution on the six bounding surfaces (we set $\xi_1 = 0$, $\xi_2 = 1$, and similarly for η_j and ζ_k). The blending functions of the linear TFI read

$$\begin{aligned}
\alpha_1^0(\xi) &= 1 - \xi\,, \quad \alpha_2^0(\xi) = \xi \\
\beta_1^0(\xi) &= 1 - \eta\,, \quad \beta_2^0(\xi) = \eta \\
\gamma_1^0(\xi) &= 1 - \zeta\,, \quad \gamma_2^0(\xi) = \zeta\,.
\end{aligned} \tag{11.5}$$

The linear TFI is computationally very efficient. An example of algebraically generated grid using the linear TFI is shown in Fig. 11.8.

The Hermite TFI with cubic blending functions allows it to prescribe additionally the slopes of the grid lines at the boundaries. The Hermite TFI results if we use $L = M = N = 2$ and $P = Q = R = 1$ in Eq. (11.2) and Eq. (11.4). However, it is also possible to mix linear and Hermite interpolations in different computational coordinates. A description of the Hermite TFI and further extensions to the TFI technique (e.g., grid spacing control) can be found in [12], Chapter 3.

11.1.3 Elliptic Grid Generation

Grid generation methods based on elliptic PDE's are known to produce grids with smoothly varying cell sizes and slopes of the grid lines. Furthermore, elliptic grid generation methods offer the possibility to control the orthogonality and the spacing near boundaries, which is particularly important for the simulation of viscous flows. Elliptic PDE's in grid generation were introduced first by Thompson et al. [16] in 1974. A detailed description of the numerical implementation in 2D was presented, e.g., in [17] and [18].

In 3D, the system of Poisson equations for the unknown Cartesian coordinates $\vec{r} = [x, y, z]^T$ of the grid points can be written as

$$\begin{aligned}
&\alpha_{11}\frac{\partial^2 \vec{r}}{\partial \xi^2} + \alpha_{22}\frac{\partial^2 \vec{r}}{\partial \eta^2} + \alpha_{33}\frac{\partial^2 \vec{r}}{\partial \zeta^2} \\
&\quad + 2\left(\alpha_{12}\frac{\partial^2 \vec{r}}{\partial \xi \partial \eta} + \alpha_{13}\frac{\partial^2 \vec{r}}{\partial \xi \partial \zeta} + \alpha_{23}\frac{\partial^2 \vec{r}}{\partial \eta \partial \zeta}\right) = \\
&\quad -\frac{1}{J^2}\left(P\frac{\partial \vec{r}}{\partial \xi} + Q\frac{\partial \vec{r}}{\partial \eta} + R\frac{\partial \vec{r}}{\partial \zeta}\right),
\end{aligned} \tag{11.6}$$

where P, Q and R denote the *control functions*. The metric coefficients α are given by the relations

$$\begin{aligned}
\alpha_{11} &= \left(\frac{\partial \vec{r}}{\partial \eta}\cdot\frac{\partial \vec{r}}{\partial \eta}\right)\left(\frac{\partial \vec{r}}{\partial \zeta}\cdot\frac{\partial \vec{r}}{\partial \zeta}\right) - \left(\frac{\partial \vec{r}}{\partial \eta}\cdot\frac{\partial \vec{r}}{\partial \zeta}\right)^2 \\
\alpha_{22} &= \left(\frac{\partial \vec{r}}{\partial \xi}\cdot\frac{\partial \vec{r}}{\partial \xi}\right)\left(\frac{\partial \vec{r}}{\partial \zeta}\cdot\frac{\partial \vec{r}}{\partial \zeta}\right) - \left(\frac{\partial \vec{r}}{\partial \xi}\cdot\frac{\partial \vec{r}}{\partial \zeta}\right)^2 \\
\alpha_{33} &= \left(\frac{\partial \vec{r}}{\partial \xi}\cdot\frac{\partial \vec{r}}{\partial \xi}\right)\left(\frac{\partial \vec{r}}{\partial \eta}\cdot\frac{\partial \vec{r}}{\partial \eta}\right) - \left(\frac{\partial \vec{r}}{\partial \xi}\cdot\frac{\partial \vec{r}}{\partial \eta}\right)^2 \\
\alpha_{12} &= \left(\frac{\partial \vec{r}}{\partial \xi}\cdot\frac{\partial \vec{r}}{\partial \zeta}\right)\left(\frac{\partial \vec{r}}{\partial \eta}\cdot\frac{\partial \vec{r}}{\partial \zeta}\right) - \left(\frac{\partial \vec{r}}{\partial \xi}\cdot\frac{\partial \vec{r}}{\partial \eta}\right)\left(\frac{\partial \vec{r}}{\partial \zeta}\cdot\frac{\partial \vec{r}}{\partial \zeta}\right) \\
\alpha_{13} &= \left(\frac{\partial \vec{r}}{\partial \xi}\cdot\frac{\partial \vec{r}}{\partial \eta}\right)\left(\frac{\partial \vec{r}}{\partial \eta}\cdot\frac{\partial \vec{r}}{\partial \zeta}\right) - \left(\frac{\partial \vec{r}}{\partial \xi}\cdot\frac{\partial \vec{r}}{\partial \zeta}\right)\left(\frac{\partial \vec{r}}{\partial \eta}\cdot\frac{\partial \vec{r}}{\partial \eta}\right) \\
\alpha_{23} &= \left(\frac{\partial \vec{r}}{\partial \xi}\cdot\frac{\partial \vec{r}}{\partial \zeta}\right)\left(\frac{\partial \vec{r}}{\partial \xi}\cdot\frac{\partial \vec{r}}{\partial \eta}\right) - \left(\frac{\partial \vec{r}}{\partial \eta}\cdot\frac{\partial \vec{r}}{\partial \zeta}\right)\left(\frac{\partial \vec{r}}{\partial \xi}\cdot\frac{\partial \vec{r}}{\partial \xi}\right).
\end{aligned} \tag{11.7}$$

In Eq. (11.7), the terms in brackets represent scalar products. The inverse of the determinant of the coordinate transformation Jacobian (J^{-1}) is evaluated according to Eq. (A.13). It should be noted that by setting $P = Q = R = 0$, Eq. (11.6) reduces to the Laplace equation. The inherent smoothing properties of the Laplace equation can be utilised to improve the quality of an algebraically generated grid. However, the point distribution in the interior field cannot be controlled.

Elliptic equations for the generation of 2-D or surface grids are easily derived from Eq. (11.6). After the multiplication with J^2 we obtain

$$\begin{aligned}
\alpha_{11}\left(\frac{\partial^2 x}{\partial \xi^2} + P\frac{\partial x}{\partial \xi}\right) - 2\alpha_{12}\frac{\partial^2 x}{\partial \xi \partial \eta} + \alpha_{22}\left(\frac{\partial^2 x}{\partial \eta^2} + Q\frac{\partial x}{\partial \eta}\right) &= 0 \\
\alpha_{11}\left(\frac{\partial^2 y}{\partial \xi^2} + P\frac{\partial y}{\partial \xi}\right) - 2\alpha_{12}\frac{\partial^2 y}{\partial \xi \partial \eta} + \alpha_{22}\left(\frac{\partial^2 y}{\partial \eta^2} + Q\frac{\partial y}{\partial \eta}\right) &= 0\,.
\end{aligned} \tag{11.8}$$

The metric coefficients in Eq. (11.8) become

$$\begin{aligned}
\alpha_{11} &= \left(\frac{\partial x}{\partial \eta}\right)^2 + \left(\frac{\partial y}{\partial \eta}\right)^2 \\
\alpha_{12} &= \frac{\partial x}{\partial \xi}\frac{\partial x}{\partial \eta} + \frac{\partial y}{\partial \xi}\frac{\partial y}{\partial \eta} \\
\alpha_{22} &= \left(\frac{\partial x}{\partial \xi}\right)^2 + \left(\frac{\partial y}{\partial \xi}\right)^2 .
\end{aligned} \tag{11.9}$$

The generation of a boundary orthogonal grid requires the specification of appropriate boundary conditions during the solution of Eq. (11.6) or Eq. (11.8). Basically, we can apply either *Neumann* or *Dirichlet* conditions. Neumann boundary conditions allow it to prescribe directly the intersection angle between a grid line and the boundary. In this case, the control functions are not required ($P = Q = R = 0$). However, the point distribution on the boundary and the spacing cannot be controlled. In fact, the boundary nodes will be automatically redistributed to match the given grid line skewness. Hence, the approach is not suited for viscous wall boundaries. More details on the Neumann conditions in elliptic grid generation are provided, e.g., in Ref. [12], Chapter 6.

Dirichlet boundary conditions are used in cases where the positions of the boundary points have to stay fixed. The control functions are then employed to achieve the desired intersection angles and spacing. The effects of the control functions on the grid are presented in Fig. 11.11 for the 2-D case. As we can see, negative values of P in Eq. (11.8) cause the lines η = const. to move in the direction of decreasing ξ, whereas $Q < 0$ shifts the lines ξ = const. to lower η-values. Thus, since the boundary nodes are fixed, varying P changes the angle between the grid lines η = const. and the boundary. The values of the control functions are calculated first at the respective boundaries from the difference between the prescribed and the actual skewness and grid spacing. Then, the boundary values of P, Q and R are interpolated in the interior of the domain and the system Eq. (11.6) or Eq. (11.8) is solved. The procedure is repeated until the required grid properties are achieved. This approach based on Dirichlet conditions was developed in 2D by Sorenson [17] and by Thomas and Middlecoff [19]. Later it was extended to 3D by Sorenson [20] and Thompson [21].

The elliptic equations (11.6) or (11.8) are usually discretised using second-order central finite differences. The resulting set of linear algebraic equations

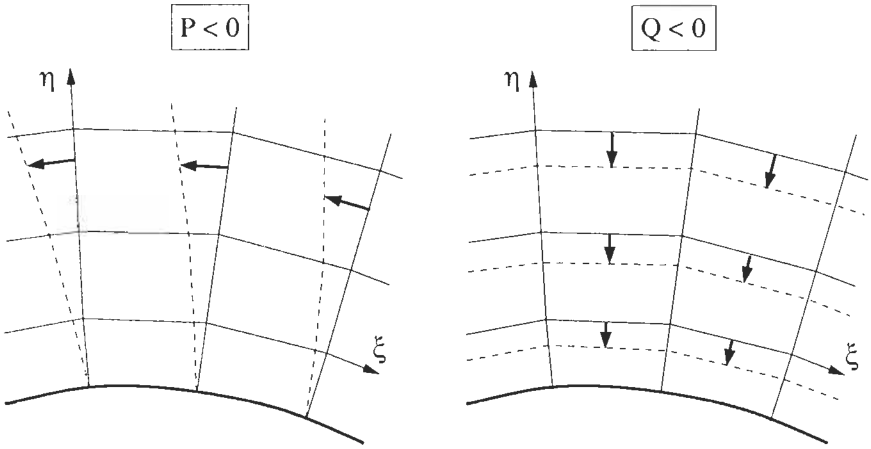


Figure 11.11: Effects of the control functions in 2-D elliptic grid generation. P controls the skewness and Q the spacing.

can be solved by any standard technique, e.g., by the Gauss-Seidel relaxation scheme accelerated by multigrid. The values of the control functions are updated in an outer iteration. In order to increase the robustness of the procedure, it is advisable to carry out several iterations with the Laplace equations ($P = Q = R = 0$), in order to smooth the initial (usually algebraic) grid [22]. Examples of elliptically generated grids are shown in Figs. 11.1, 11.2 and 11.6.

11.1.4 Hyperbolic Grid Generation

Hyperbolic PDE's are suitable for grid generation in cases where the shape of the outer boundary need not to be exactly controlled. In order to generate the grid, an initial point distribution has to be prescribed. Then, the grid is build by marching a given distance in the normal direction from a known to a new layer of grid points. The application of hyperbolic PDE's for grid generation was proposed by Starius [23], and by Steger and Chaussee [24]. A recent discussion of the hyperbolic grid generation methodology can be found in [12], Chapter 5. Various extensions and improvements were described in Refs. [25]-[30].

The hyperbolic equations for the generation of a volume grid read

$$
\begin{aligned}
\frac{\partial \vec{r}}{\partial \xi} \cdot \frac{\partial \vec{r}}{\partial \zeta} &= 0 \\
\frac{\partial \vec{r}}{\partial \eta} \cdot \frac{\partial \vec{r}}{\partial \zeta} &= 0 \\
\frac{\partial \vec{r}}{\partial \zeta} \cdot \left(\frac{\partial \vec{r}}{\partial \xi} \times \frac{\partial \vec{r}}{\partial \eta} \right) &= \Omega \,,
\end{aligned}
\tag{11.10}
$$

where $\vec{r} = [x, y, z]^T$ denotes the Cartesian coordinates of the grid points, ξ, η, ζ stand for the computational coordinates (Eq. (11.1)), and Ω is the user-specified cell volume. Furthermore, we assumed in Eq. (11.10) that the surface ζ = const. represents the initial state. The first two relations in Eq. (11.10) represent orthogonality conditions (ξ = const. and η = const. orthogonal to ζ = const. plane). The last relation in Eq. (11.10) guarantees that the cell volume becomes equal to Ω. This opens the possibility to control the spacing between the grid layers.

The 2-D formulation of the hyperbolic equations can be written as

$$\begin{aligned} \frac{\partial x}{\partial \xi}\frac{\partial x}{\partial \eta} + \frac{\partial y}{\partial \xi}\frac{\partial y}{\partial \eta} &= 0 \\ \frac{\partial x}{\partial \xi}\frac{\partial y}{\partial \eta} - \frac{\partial y}{\partial \xi}\frac{\partial x}{\partial \eta} &= \Omega\,, \end{aligned} \tag{11.11}$$

where Ω denotes now the prescribed cell area. The initial point distribution is given here on the curve $\eta = 0$.

The hyperbolic grid generation equations (11.10) or (11.11) are discretised with respect to the ξ and η coordinate (in 3D) or the ξ coordinate (in 2D) using second-order central differences. The marching in the ζ-direction (Eq. (11.10)) or in the η-direction (Eq. (11.11)) is carried out by a first-order implicit scheme (ADI scheme – see Subsection 6.2.3). In this way, the marching step can be selected based only on the desired grid spacing. The implicit operator is inverted using a standard tridiagonal solver. Artificial dissipation terms (second differences) have to be added to the discretised equations in order to stabilise the marching procedure. The solution of the hyperbolic equations (11.10) or (11.11) is faster and usually also easier than that of the elliptic system from the previous subsection.

11.2 Unstructured Grids

Unstructured grids are typically composed of triangles in 2D and of tetrahedra in 3D. However, nowadays it becomes increasingly popular to build unstructured grids from various element types. For example, hexahedra or prisms are employed to discretise boundary layers. The rest of the flow domain is filled with tetrahedra. Pyramids are used as transitional elements between the hexahedra or the prisms and the tetrahedra. Hence the name *mixed element* grids. The advantages of structured hexahedral grids are the preserved accuracy in the wall normal direction for highly stretched viscous grids as well as the reduced number of elements, edges and faces as compared to a tetrahedral grid. On the other hand, the desirable feature of unstructured tetrahedral grids is the capability to discretise complex geometries (like in Figs. 11.3, 11.4) fast and with a minimum user intervention. Mixed grids seek to combine the advantages of both approaches.

In the case of unstructured grids, nodes and grid cells are quasi randomly ordered, i.e., neighbouring cells or grid points cannot be directly identified by their indices (cf. Fig. 3.3). This leads to tremendous geometric flexibility of unstructured grids, since the grid does not need to conform to any predetermined topology. Furthermore, adaptation of the grid to the physical solution – grid refinement or coarsening – is much easier to accomplish on unstructured than on structured grids.

Unstructured grid generation methodologies for CFD applications are mostly based on either an

- *Delaunay*, or
- *advancing-front*

method. Both approaches can also be combined together. Depending on the base methodology, we speak either of *advancing-front Delaunay* [31] or of *frontal Delaunay* schemes [32]-[34]. Besides the both standard techniques, there are also rather new and interesting methods like the so-called *bubble packing* algorithm [35]-[37]. Here, we shall describe only the basic features of the Delaunay and the advancing-front method. A survey of both approaches is contained, e.g., in Refs. [38], [39] and [12].

The Delaunay approach primarily refers to a particular way of connecting grid points to form triangles in 2D and tetrahedra in 3D. The most important feature of the Delaunay triangulation is that the circumcircle (or the circumsphere in 3D) of any triangle (tetrahedron) contains no other grid point. The consequence of the empty circumcircle criterion is that in 2D the minimum angle is maximised for all triangles (*max-min triangulation*). Thus, a high grid regularity can be achieved.

The idea of the advancing-front method is to generate the grid sequentially element by element starting from a known boundary discretisation (surface grid). The open surfaces of the elements constitute the front. New triangles (tetrahedra) are constructed by placing points ahead of the front. In this way,

the front moves through the domain until all cavities are filled. The point placement is controlled by the so-called *background grid*. The advancing-front approach offers the advantages of smooth point distribution and implicitly assured boundary integrity. However, it is slower than the Delaunay method.

11.2.1 Delaunay Triangulation

The Delaunay triangulation is based on a methodology proposed by Dirichlet [40] in 1850 for the unique subdivision of space into a set of packed convex regions. Given a set of points, each region represents the space around the particular point, which is closer to that point than to any other. The regions form polygons (polyhedra in 3D) which are known as the *Dirichlet tessellation* or the *Voronoj diagram* [41]. If we connect point pairs which share some segment (face) of the Voronoj diagram by straight lines, we obtain the Delaunay triangulation [42]. The triangulation defines a set of triangles (tetrahedra in 3D), which cover the convex hull of the points. This is displayed in Fig. 11.12. The Delaunay triangulation is the dual of the Voronoj diagram. The nodes of the Voronoj polygons are in 2D the centres of circumcircles of the triangles. In 3D, the nodes represent the centres of circumspheres of the tetrahedra. This implies that the circumcircle of every triangle (circumsphere of every tetrahedron) contains no point from the set in its interior.

As we already stated, the Delaunay method represents a particular way of connecting grid points. The positions of the points must be determined by some other technique. Therefore, one popular approach for the construction of a De-

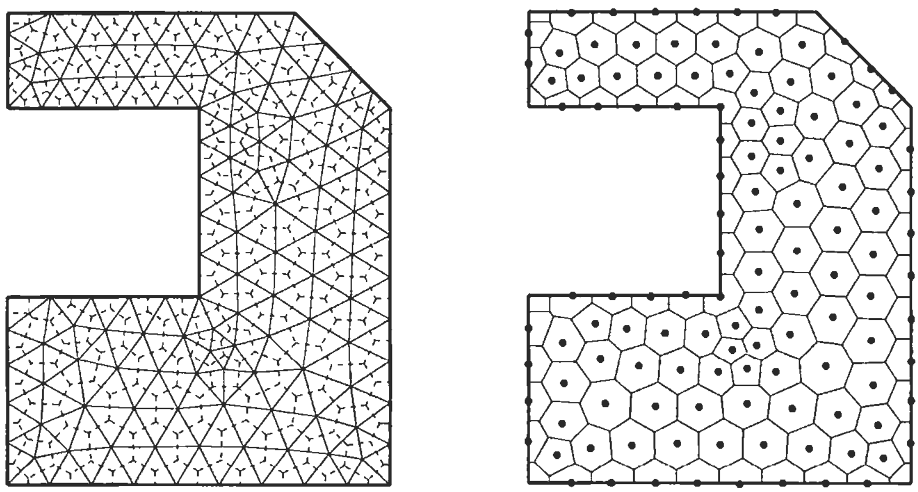


Figure 11.12: The Delaunay triangulation (left) and the Voronoj diagram (left as dashed line, right as solid line).

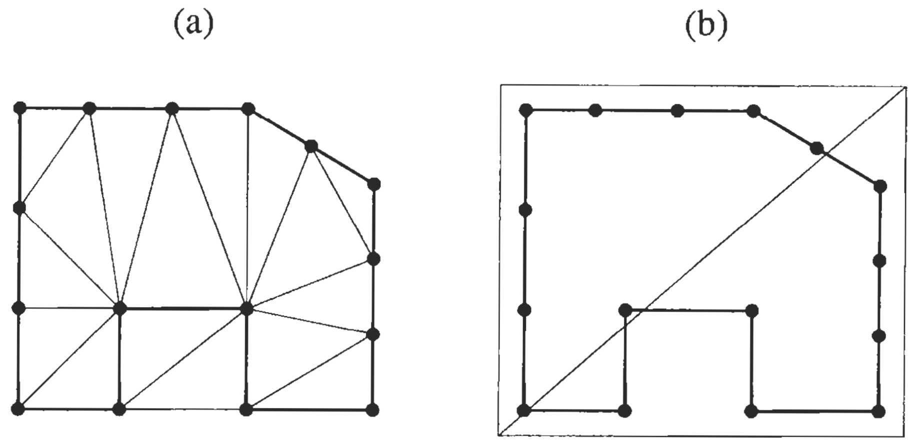


Figure 11.13: Possibilities for initial Delaunay grid in 2D: triangulation of boundary nodes (a); quadrilateral divided into two triangles (b).

launay grid is to insert sequentially nodes into an initial triangulation. The grid is then locally retriangulated in order to fulfil the empty circumcircle (circumsphere) criterion. The *incremental point-insertion* strategy can be described by the following steps:

1. discretise the boundaries of the physical domain.
2. Generate an initial Delaunay grid which covers all boundary nodes. This can be either a triangulation of the boundary nodes itself (Fig. 11.13a) or a surrounding quadrilateral (hexahedron in 3D), which is decomposed into triangles (tetrahedra) as displayed in Fig. 11.13b.
3. If not already done, insert all boundary nodes into the initial triangulation using a Delaunay-conforming technique.
4. Build a list of all triangles (tetrahedra) in the grid which violate some size or quality measure. Order the list to start with the worst element.
5. Place a new point at the circumcentre [43]-[45] (see Fig. 11.14) or at the Voronoj segment [46], [47] of the first element in the list and locally retriangulate the grid. Check each new element and add it to the list if not in accordance with the size/quality measure.
6. If there are still elements in the list, go to step 5.
7. Delete elements outside of the domain and recover the boundaries.
8. Check the grid quality (Subsection 11.2.5). Smooth the grid and/or swap edges if necessary.

Alternatively, boundary recovery (step 7) can be carried out after the step 3.

As we can see, the core task of the Delaunay-grid generation is the insertion of a new point into a valid triangulation (steps 3 and 5). Several algorithms were proposed for this purpose. The most popular approaches are due to Green and Sibson [48], Bowyer [49], and Watson [50]. We shall describe Watson's algorithm further below.

In order to accomplish the steps 4 and 5, we have to assess the elements with respect to some appropriate measure. This can be either the quality of the element (minimum angle, aspect ratio – see [51], [52]), or the size (volume, edge length) of the element. The quality of an element can be assessed directly from its geometry. However, the size measure has to be formulated as a function of the spatial position within the domain. Several approaches are possible:

- specification of an analytical function (e.g., size is proportional to distance from the body);
- interpolation of size distribution from boundaries using the initial triangulation (see, e.g., [12], Ch. 1, pp. 17–20);
- interpolation on background grid based on quadtree (octree in 3D) structure [53]-[57];
- specification of sources (point, line, etc.) inside the domain (see, e.g., [12], Ch. 1, pp. 20–22).

Furthermore, the points can be placed with the aid of the advancing-front technique [32]-[34].

Watson Algorithm

The point insertion and retriangulation method of Watson consists of the following steps [50]:

1. locate the element which contains the inserted point P (Fig. 11.14).
2. Find all elements whose circumcircle (circumsphere in 3D) is intersected by point P. This situation is sketched on the left-hand side of Fig. 11.15.
3. Delete all intersected elements from the triangulation.
4. Form new elements by connecting the points on the boundary of the convex cavity to the point P (right-hand side of Fig. 11.15).

Data structure particularly suitable for the algorithm of Watson results when we store for each element:

- indices of the forming nodes;
- pointers to neighbours which share a common face with the element;
- circumcentre and radius of the circumcircle (circumsphere).

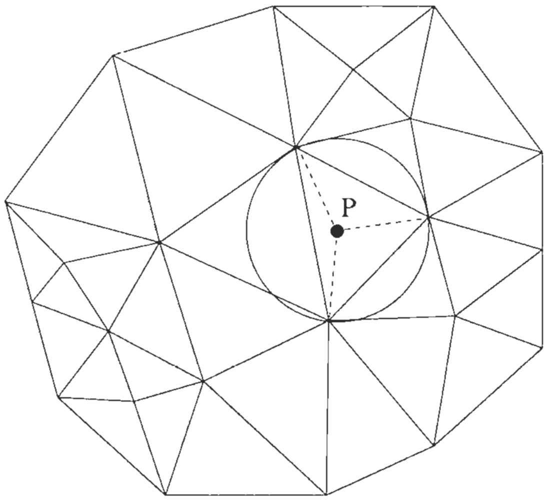


Figure 11.14: Insertion of new node P into valid Delaunay triangulation. The node is located at the centre of circumcircle.

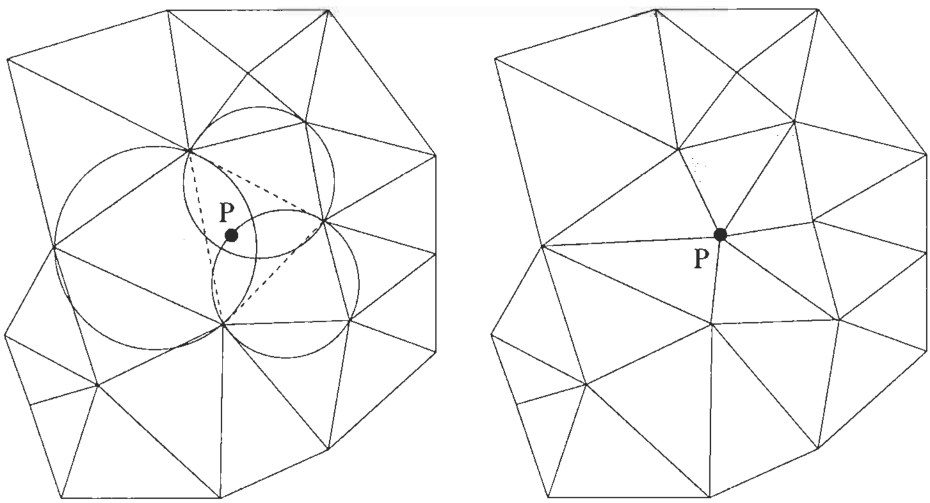


Figure 11.15: Watson algorithm: deletion of invalid triangles (left) and retriangulation of the convex cavity (right).

This data structure allows for an efficient search of intersected elements in the step 2. We start the search with the neighbours of the element which contains the inserted point. Then, we proceed to the neighbours of these neighbours and so on. We stop the search in a particular direction, if the circumcircle (circumsphere) of the element is not intersected. The geometrical properties of the Delaunay triangulation make sure that the neighbours of this element are not intersected. It is also guaranteed that all elements being involved are localised by this simple strategy [50], [58]. The numerical effort of Watson's algorithm is of the order of $N \log(N)$, where N is the number of grid points. This results in a very fast grid generation methodology.

Constrained Delaunay Triangulation

The Delaunay triangulation does not automatically take care of prescribed edges and faces, like those on the boundaries of the physical domain. This is the purpose of the so-called *constrained Delaunay* triangulation [59]. The restoration of boundary edges in 2D is sketched in Fig. 11.16. Depending on the situation, either edge swapping or retriangulation is required. The constrained Delaunay triangulation leads in 2D always to a valid grid. However, this cannot be guaranteed in 3D. The recovery of the boundary discretisation in 3D has to be conducted in two steps - first for boundary edges and second for boundary faces [39]. In some cases, additional points (so-called *Steiner points*) have to be inserted on the boundaries [60]-[62]. Otherwise, the cavity cannot be retetrahedralised. After the boundary edges (and faces) are recovered, the elements outside of the flow domain can be deleted.

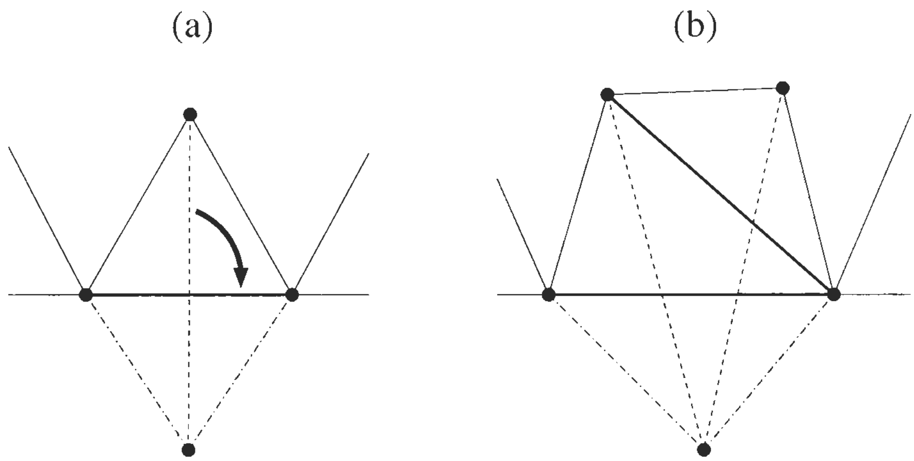


Figure 11.16: Insertion of missing boundary edge: by edge swapping (a); by deleting intersected triangles and retriangulating the cavity (b). Swapped or deleted edges are represented by dashed lines, new edges by thick solid lines. The triangles outside of the domain (dash-dotted line) are removed.

11.2.2 Advancing-Front Method

The advancing-front methodology was first introduced by Peraire et al. [63], [64], and by Löhner et al. [65] by the end of the 1980's. The individual steps of this grid generation scheme can be summarised as follows:

1. discretise the boundaries of the physical domain (generate surface grids).

2. Generate a list of edges (faces in 3D) which represent the front. Initially, these are the boundary edges (faces). Sort the list in the order of increasing edge (face) size [38]. This strategy helps to generate smoothly varying elements.

3. Select the first edge (face) of the list and place a new point P_0 in the normal direction above the centre of the front edge (face). The situation is displayed in Fig. 11.17. The distance d into the domain is governed by the local values of the size-distribution function (see, e.g., [39]).

4. Define a circle (sphere) with radius r centred at the point P_0. The radius r depends on the local grid size.

5. Determine all points which are located within the circle (sphere).

6. If there are no intersected points, generate a new element with the point P_0. Otherwise, order the intersected points with respect to their distance to P_0 (i.e., P_1, P_2, P_3). Form elements with the points and accept the first one which does not intersect any other element and satisfies given quality measure(s).

7. Delete the current front edge (face) and add the newly formed edges (faces) to the list. Sort the list again.

8. Continue with step 3 until the list is empty.

9. Check the quality of the grid (see Subsection 11.2.5). Smooth the grid and/or swap edges if required.

Different stages of the advancing-front process are shown in Fig. 11.18a-d for an exemplary 2-D configuration.

The distribution of the element size in the domain is mostly governed by the point density on the boundaries. Additionally, the distance d in step 3 can be controlled by placing sources of various type – point, line, cylinder, etc. – inside the domain (see, e.g., [12], Ch. 1, pp. 20–22). The local element size can be obtained from a background grid based on the quadtree (octree in 3D) data structure [53]-[57]. Another possibility is to interpolate the sizes by employing the Delaunay triangulation of the boundary nodes [32]-[34], [66], or to use a Cartesian background grid [67]. The quadtree (octree) data structure can also be used to search efficiently for nearby points (step 5) as well as to check for possible intersections between the elements. A simple test for the intersection

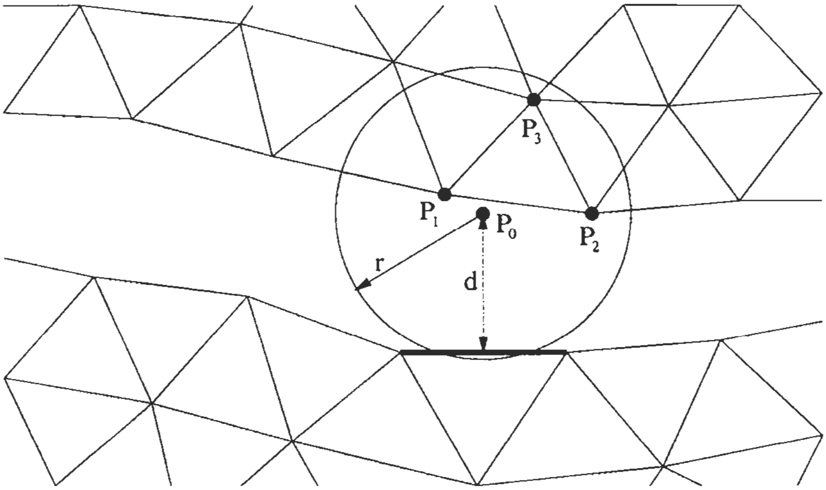


Figure 11.17: Insertion of a new node P_0 in the 2-D advancing-front method. The active edge is denoted by a thick line.

between fronts based on spring analogy was devised in [68] and [3]. A detailed comparison of different search algorithms can be found, e.g., in [69].

The advantages of the advancing-front method as compared to the Delaunay triangulation scheme are the implicitly retained boundary discretisation and the better control over element size and grid smoothness. Furthermore, the Delaunay approach is less robust (more sensitive to round-off errors) than the advancing-front method. However, the point searching and intersection checking algorithms require significant numerical effort. Today, a combination between the advancing-front and the Delaunay methodology is often employed in CFD, particularly in 3D [31]-[34], [70].

11.2.3 Generation of Anisotropic Grids

In the preceding Subsections 11.2.1 and 11.2.2, we described two methodologies for the generation of **isotropic** triangular or tetrahedral elements. This is appropriate for the simulation of inviscid flows. However, an accurate solution of the Reynolds-averaged Navier-Stokes equations requires high spatial resolution in the direction across boundary layers and wakes. Thus, in the case of high Reynolds-number flows, isotropic grid would result in an excessively large number of elements. Clearly, we have to employ **anisotropic**, high aspect-ratio elements in the viscous flow regions. Basically, we can utilise quadrilaterals (hexahedra or prisms in 3D), which allow quite naturally for stretching. This leads to mixed-element grids which are very popular today (see next subsection). Of course, it is possible to decompose the hexahedra or prisms in order to obtain a purely tetrahedral grid. On the other hand, we can generate di-

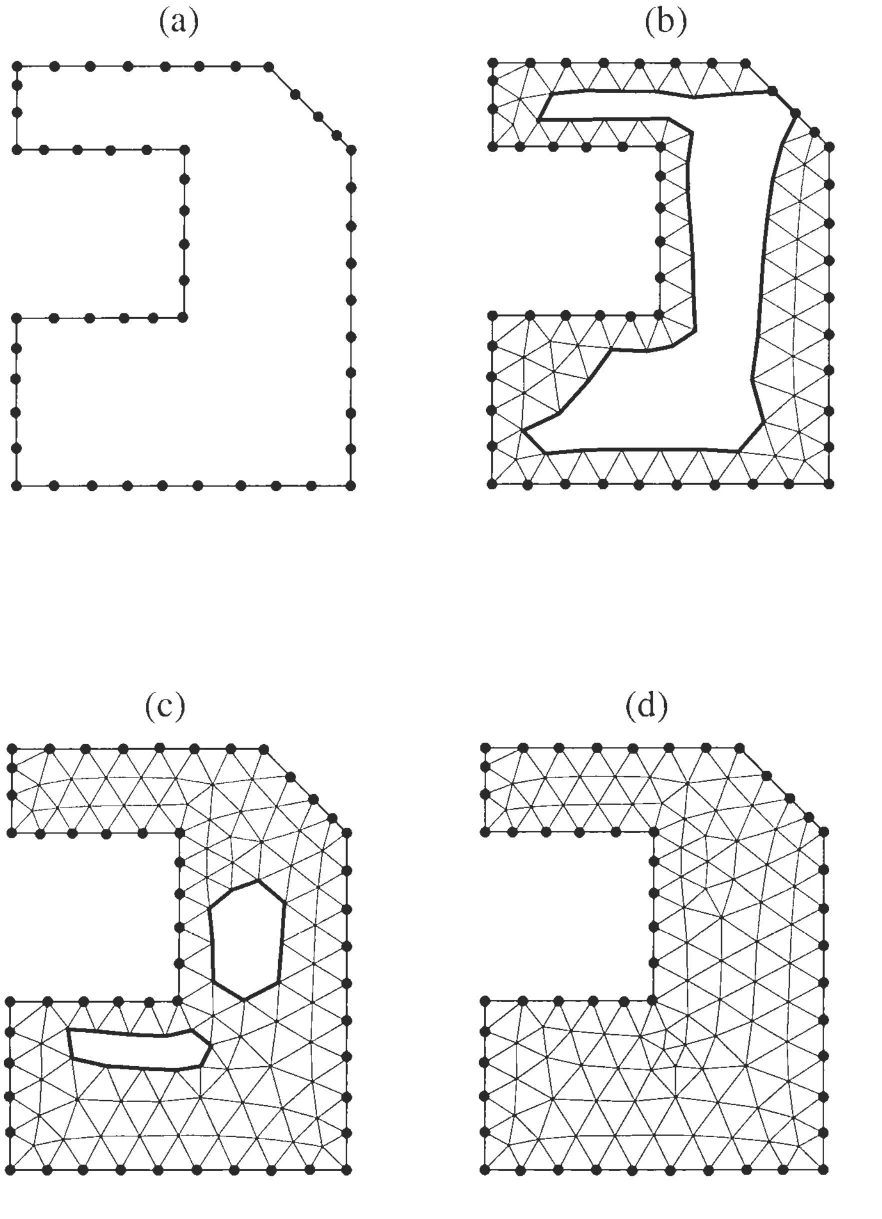


Figure 11.18: Different stages of grid generated using the advancing-front technique. The actual front is represented by a thick solid line.

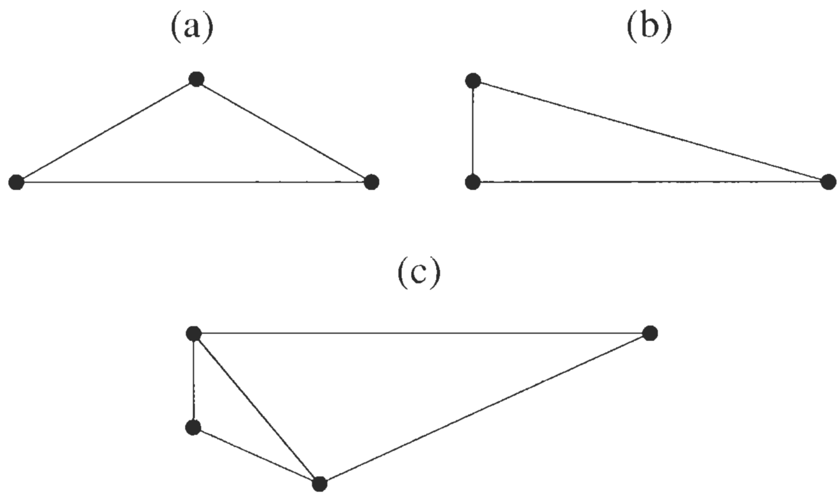


Figure 11.19: High aspect-ratio elements: obtuse triangle (a), right-angle triangle (b), and right-angle tetrahedron (c).

rectly stretched triangular (tetrahedral) grids. In any case, the grid generation scheme should prevent the creation of obtuse triangles (tetrahedra) of the form sketched in Fig. 11.19a. It was demonstrated in [71] that such elements lead to exceedingly high truncation error of the spatial discretisation. Therefore, it is suggested to generate right-angle elements like those displayed in Fig. 11.19b and 11.19c. However, depending on the type of the control volume, right-angle elements may also induce discretisation errors [72], [73] (cf. Subsection 5.2.3).

Stretched Delaunay triangulation

The generation of stretched triangular (tetrahedral) elements requires the definition of the magnitude and the direction of the stretching in the physical domain. Background grid has to be used for this purpose in the case of stretched Delaunay triangulation. The same point insertion techniques can be employed as for the generation of isotropic Delaunay grids. However, the circumcircle (circumsphere) criterion is replaced by a condition of empty circumellipse (cirumellipsoid). The ellipse (ellipsoid) is oriented in the local direction of the stretching vector and the magnitude of stretching is reflected by the ratio of the axes [74]-[76].

Advancing-Layers Method

The specification of the stretching vector in the anisotropic Delaunay triangulation is quite involved for geometrically complex domains. Therefore, the so-called *advancing-layers* method proposed by Pirzadeh [77], [78], [3], [4] and others [79], [80], is more widely utilised. The advancing-layers method is similar

to the hyperbolic structured grid generation technique (Subsection 11.1.4). It can also be considered as a modified advancing-front technique.

The generation of stretched triangular (tetrahedral) grids by the advancing-layers approach proceeds according to the following steps (see Fig. 11.20):

1. triangulate all boundary surfaces.

2. Compute approximate normal vectors at boundary nodes.

3. Place grid points along the surface normals and form layers of quadrilateral (prismatic in 3D) elements of increasing thickness.

4. Decompose each quadrilateral (prism) into two triangles (three tetrahedra). The connectivity pattern requires particular attention in 3D.

5. Continue growing each stack of elements until the front intersects either itself or another front, or until the cell aspect-ratio becomes close to unity.

The rest of the flow domain is then filled with isotropic tetrahedra. Usually, the advancing-front methodology is employed, which starts from the surface represented by the last layer of boundary cells. Figure 11.4 shows an example of a grid generated using the above procedure.

In principle, two options exist for the evaluation of the normal vectors at the boundary nodes (step 2). First, if the bounding surfaces are described by

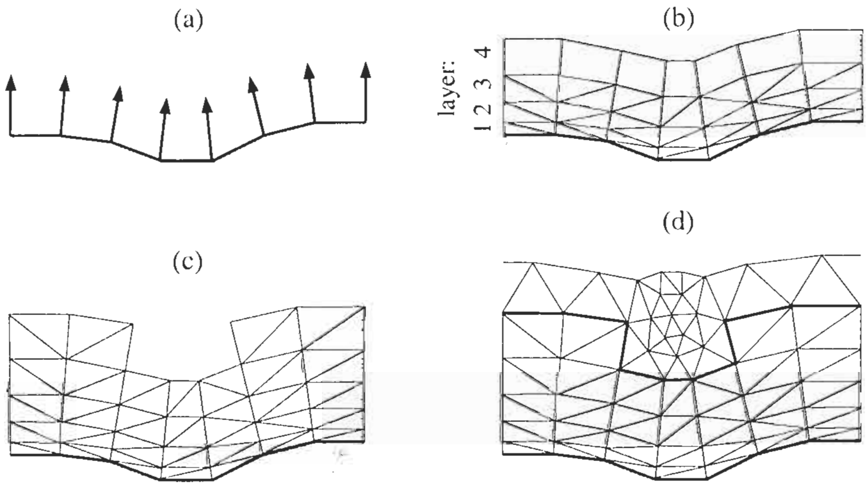


Figure 11.20: Steps of the advancing-layer method: computation of boundary normals (a), generation of a new layer and subdivision of the elements (b), finished stretched grid (c), generation of isotropic elements in the rest of the domain (d).

analytical functions like NURBS (Non-Uniform Rational B-Splines) [81], the normal derivatives are easily calculated. A grid generation method of this type was described in Ref. [82]. However, care has to be taken at patch interfaces (e.g., like at trailing edges).

The second, more widespread approach, is to use the surface triangulation. This can be done quite easily in 2D, where it is sufficient to average the normals of the boundary edges which join at the particular node. However, such simple averaging is not appropriate in 3D. The reason is that the boundary node can be shared by any number of triangular faces. Hence, the normal vector can become biased depending on the triangulation. The problem is particularly severe at sharp corners and bends. The necessary condition for an optimal normal vector is given by the so-called *visibility criterion* [83], [84]. It states that a point placed along the normal direction has to be equally visible from all triangular faces sharing the boundary node. Various procedures based on the visibility criterion were devised for the construction of the normal vectors. For details see, e.g., Refs. [83]-[86], [3]. The normal vectors are smoothed, in order to prevent abrupt changes of the marching direction and crossing of the grid lines. In general, weighted Laplacian type smoothing is employed [84], [85], i.e.,

$$\vec{n}_i = \omega\,\vec{n}_i^{(0)} + \frac{(1-\omega)}{\sum_j 1/|\vec{r}_{ij}|}\sum_j \frac{\vec{n}_j}{|\vec{r}_{ij}|}\,. \tag{11.12}$$

In above Eq. (11.12), $\vec{n}_i$ and $\vec{n}_i^{(0)}$ denote the new and the initial node-normals, respectively. Furthermore, $\vec{n}_j$ represents the normal vectors at the adjacent nodes and $|\vec{r}_{ij}|$ is the distance between the nodes i and j. The weighting factor ω depends on the surface curvature. It takes small values in concave regions and large values in convex ones. The smoothing is applied separately for each layer. A common procedure is to use the initial node-normals for the first few layers and then to smooth increasingly the normal vector. The reason is that a boundary orthogonal grid helps to reduce the numerical error.

The size of the marching step is calculated based on a user-specified thickness of the first layer and on stretching rate in the normal direction. Hence, for example, the spacing can be obtained from [3]

$$\Delta n_k = \Delta n_0 \left[1 + a(1+b)^{k-1}\right]^{k-1}\,, \tag{11.13}$$

where Δn_k represents the spacing of the k-th layer, Δn_0 is the given first-layer thickness, and $0.04 \le a \le 0.2$ and $0 \le b \le 0.07$ stand for the stretching parameters. The marching step Δn_k can be additionally modified at convex and concave corners [86]. It is also possible to increase Δn_k in the direction of growing boundary layer to keep y^+ constant.

Each stage of the advancing-layers algorithm generates in 2D a layer of quadrilaterals and in 3D a layer of prisms. Whereas the quadrilaterals can be easily divided into two triangles (along the shortest diagonal), the decomposition of prisms into three tetrahedra requires some care. The point is that faces between the prisms have to be divided in the same way. An integer based

approach for the consistent decomposition of prisms was suggested in [3]. A simpler scheme was presented in [87]. A further decomposition methodology was reported in [88].

Combined Advancing-Normal/Delaunay Method

A further possibility for the generation of stretched triangular (tetrahedral) grids is offered by a modified advancing-front method (called here *advancing-normal point placement*) combined with the Delaunay triangulation. Corresponding approach was proposed by Müller [89] for 2-D grids. Marcum [90], [91] as well as Sharov and Nakahashi [86] developed 3-D versions of the algorithm. The procedure starts with the volume triangulation of the boundary nodes. The triangulation is employed as a background grid. It also serves as an efficient search structure. The boundaries can be recovered [86], but it is not necessary at this stage [90], [91]. Next, points are inserted in regions of the stretched grid using techniques similar to the advancing-layers method. The background triangulation is utilised for determination of nearby fronts and for intersection checking. In the last stage, the inflated surface is employed as a new boundary for the generation of isotropic elements by any standard Delaunay technique.

Stretched Surface Grids

In regions, where the flow exhibits a strong directionality, anisotropic grids can substantially reduce the number of triangular faces and hence volume elements without impairing the solution accuracy (see Fig. 11.21). For example, a simulation with RANS does not require high grid resolution in the spanwise direction of a wing or a blade. The same holds for a fuselage in the axial direction. Savings in the number of surface triangles between factor 2 and 6 were reported in [92], [93]. However, as indicated in Fig. 11.21, it is important to orient the surface elements properly, particularly in areas of high curvature. Otherwise, the true surface contour becomes poorly represented and the flow solution will be falsified. Stretched surface grids can be generated by any of the above methodologies for anisotropic grids (for a general discussion of surface grid generation see, e.g., [8], [94]). The stretching ratio and orientation is usually specified through a line source. Another possibility consists of generating first an isotropic grid in parametric coordinates. Then, the grid is transformed into the physical space by using stretching functions.

11.2.4 Mixed-Element/Hybrid Grids

Nowadays, it becomes increasingly popular to discretise the flow domain by using grids, which consist of different elements (prisms, tetrahedra, pyramids, etc.). Such grids are referred to as *mixed* grids or *mixed-element* grids. Another idea is to compose the grid of structured and unstructured zones [95]-[100]. In this case we speak of *hybrid* grids. It should be noted that some authors use the term "hybrid grids", but in reality they mean mixed grids. The motivation

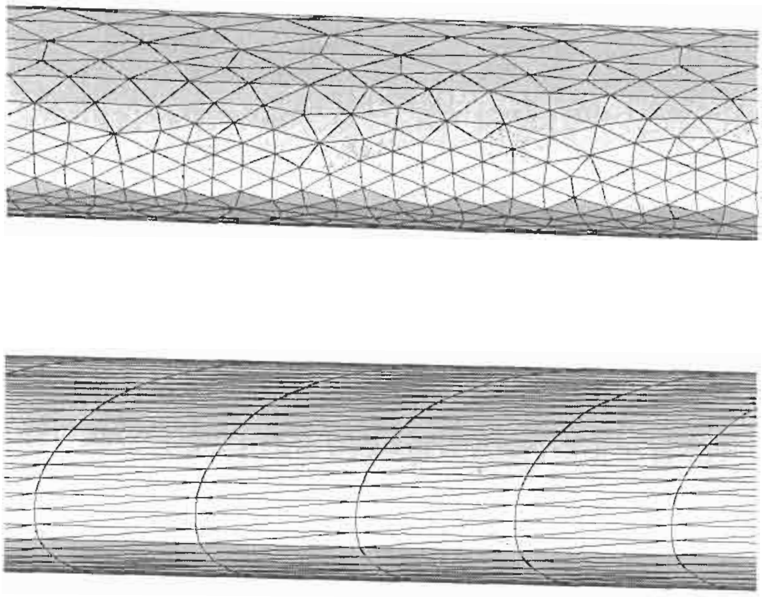


Figure 11.21: Unstructured surface grid at leading edge: isotropic with no specific orientation (top); stretched and oriented in spanwise direction (bottom).

behind the use of either the mixed or the hybrid grids is to employ the best suitable elements or grid topology in each flow region. The aims are to increase the accuracy of the simulation and to reduce the computational time. However, the constraint is that the generation of such a grid has to proceed in largely automatic way. It has also to be sufficiently fast.

Hybrid grids are only seldom employed in practice. The problem is that two different flow solvers are needed – one structured and one unstructured, which has to be coupled. The effort required not only to write, but primarily to update and to maintain two codes is significant. The classical application of hybrid grids consists of the discretisation of the near-wall regions using structured grids (hexahedral or possibly semi-structured prismatic grids), which allow for an easier implementation of higher-order spatial schemes, implicit methods [101], or multigrid [102], [103]. The rest of the domain is filled with unstructured grid, which offers considerable advantages in geometrically complex areas. The idea of hybrid grids can also be utilised in the case of rotor–stator interaction, like encountered in turbomachinery. Here, the interface between the structured grids around the rotor and the stator consists of a slice of unstructured grid (similar to the idea in [104]). The unstructured grid is re-generated with each rotor movement. In this way, the accuracy and the conservation properties of the discretisation scheme are retained across the interface.

Mixed grids offer a larger flexibility than the hybrid grids, especially for

complex geometries. Mixed grids can also be more easily generated and refined. However, the challenge is to develop data structures and numerical schemes for the flow solver, which can handle varying element types in a seamless way. We discussed some of the associated problems in Chapter 5. In the following, we briefly present methodologies for the generation of mixed prismatic/tetrahedral and prismatic/Cartesian grids.

Mixed Prismatic/Tetrahedral Grids

A method for the generation of grids consisting of prismatic elements in the near-wall region and of tetrahedral elements elsewhere was initially developed by Nakahashi [105]-[107]. The methodology was further pursued by Kallinderis [108], [109], and by Connell and Braaten [87]. The generation of the prismatic cells is conducted using the same techniques as described previously for the advancing-layer or the advancing-normal approach, respectively. The tetrahedra are grown directly from the last layer of prisms. In cases where one of the quadrilateral faces of a prism remained exposed, pyramid is used as transition element to the tetrahedra.

Mixed Prismatic/Cartesian Grids

Cartesian grids are composed of squares (cubes in 3D), which are aligned with the Cartesian coordinate axes. Cartesian grids can be generated easily and with low computational effort even for geometrically complex domains. However, the weakness of the Cartesian methods is the accuracy of the flow solution at solid boundaries, which are either curved or not oriented in the Cartesian coordinates. This problem become especially serious in the case of RANS simulations, where highly stretched and boundary orthogonal cells are required. Therefore, various authors, e.g., Melton et al. [110], Karman [111], Smith and Leschziner [112], Wang et al. [113], and Delanaye et al. [114], proposed to insert body-conforming quadrilaterals (prisms) near wall surfaces. It is possible to create a continuous interface between the inner layers and the outer Cartesian grid by using pyramids and tetrahedra as transitional elements. On the other hand, the grid generation procedure can be simplified by allowing for hanging nodes and lines. This situation is displayed in Fig. 11.22. If correctly treated, the interchange of fluxes at the interface can be kept conservative [115], [114].

11.2.5 Assessment and Improvement of Grid Quality

The quality of the grid strongly influences the accuracy of the simulation. This is particularly true for unstructured grids. We mentioned this problem in various places of Chapter 5 (e.g., in Subsection 5.3.3 on solution reconstruction). Therefore, it is important that the resulting grid consists of elements which are as regular as possible. Furthermore, the cell size (and stretching in viscous regions) should vary smoothly over the domain. Of course, the grid should also be fine enough to resolve the relevant flow features.

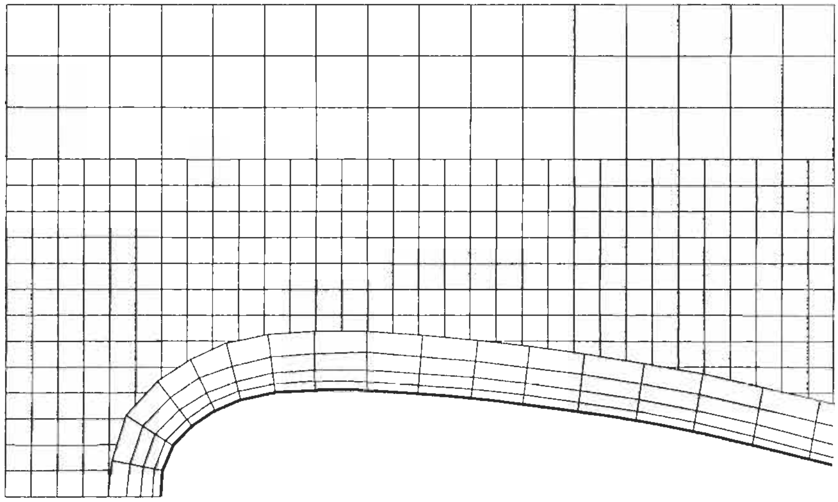


Figure 11.22: Mixed Cartesian/quadrilateral grid near boundary surface.

Figure 11.23 shows the shapes of tetrahedra, which should be avoided in the grid: obtuse elements (see also Fig. 11.19a), slivers (elements with four nearly co-planar points), needles and wedges. The only exception are wedge-like stretched tetrahedra in viscous regions. They can only be avoided by using prisms. However, since the largest gradient of the flow is in the wall-normal direction (short side of the wedge), the numerical error is not disturbing.

Beyond the element angles [91], the following parameters can be employed for tetrahedral grids as quality measures [116], [12] (number means value for an equilateral tetrahedron):

1. radius of circumscribed sphere / radius of inscribed sphere = 3.0

2. maximum edge length / radius of inscribed sphere = 4.8990

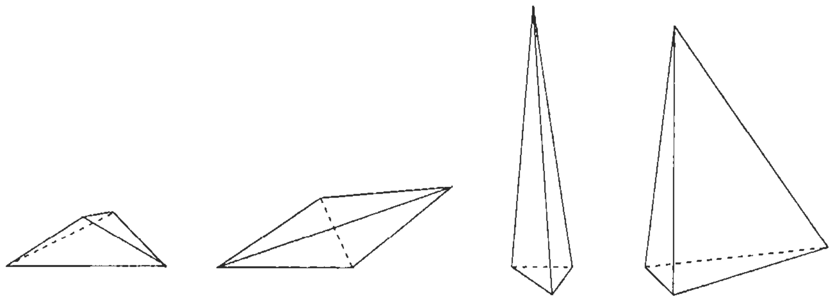
3. radius of circumscribed sphere / maximum edge length = 0.6125

4. maximum edge length / minimum edge length = 1.0

5. (average element edge length)^3/ volume = 8.4797

6. (volume)^4/ (sum of areas of all triangular faces)^3 = 4.585E-4 .

In order to improve the grid quality, the edge swapping algorithm due to Lawson [117], [118] can be used. Edge swapping is particularly suitable for removing sliver elements from the grid [34].



(a) obtuse (b) sliver (c) needle (d) wedge

Figure 11.23: Undesirable shapes of tetrahedral elements.

Grid Smoothing

Another technique, which is used quite often in order to improve the grid regularity, is smoothing. Here, the grid nodes are moved by using an approximate Laplacian operator

$$\vec{r}_i^{\,n+1} = \vec{r}_i^{\,n} + \frac{\omega}{N_A} \sum_{j=1}^{N_A} (\vec{r}_j - \vec{r}_i^{\,n}) \,, \tag{11.14}$$

where N_A denotes the number of nodes adjacent to i and ω is the relaxation factor (0.5–1.0 in the interior, 0.25 at points adjacent to boundaries, 0.0 at boundary nodes – cf. Ref. [34]). The relation in Eq. (11.14) has to be solved iteratively. The point Gauss-Seidel scheme was preferred in [33] over the point Jacobi scheme because of the better control over negative volumes.

Bibliography

[1] Brodersen, O.; Hepperle, M.; Ronzheimer, A.; Rossow, C.C.; Schöning, B.: *The Parametric Grid Generation System MegaCads.* Proc. 5th Int. Conf. Numerical Grid Generation in Comput. Field Simulations, NSF Eng. Center, Mississippi, 1996, pp. 353-362.

[2] Brodersen, O.; Ronzheimer, A.; Ziegler, R.; Kunert, T.; Wild, J.; Hepperle, M.: *Aerodynamic Applications using MegaCads.* Proc. 6th Int. Conf. Numerical Grid Generation in Comput. Field Simulations, Univ. of Greenwich, London, England, 1998, pp. 793-802.

[3] Pirzadeh, S.: *Three-Dimensional Unstructured Viscous Grids by the Advancing-Layers Method.* AIAA Journal, 34 (1996), pp. 43-49.

[4] Pirzadeh, S.: *Progress Toward a User-Oriented Unstructured Viscous Grid Generator.* AIAA Paper 96-0031, 1996.

[5] Mavriplis, D.J.; Pirzadeh, S.: *Large-Scale Parallel Unstructured Mesh Computations for 3D High-Lift Analysis.* AIAA Paper 99-0537, 1999.

[6] Pirzadeh, S.: *Unstructured Grid Generation for Complex 3D High-Lift Configurations.* Paper No. 1999-01-5557, World Aviation Congress and Exposition, San Francisco, October 1999.

[7] Reed, K.: *The Initial Graphics Exchange Specification (IGES) Version 5.1.* September 1991; see: `https://www.uspro.org`.

[8] Aftosmis, M.J.; Delanaye, M.; Haimes, R.: *Automatic Generation of CFD-Ready Surface Triangulations from CAD Geometry.* AIAA Paper 99-0776, 1999.

[9] Bohn, J.W.; Zozny, M.J.: *Automatic CAD-Model Repair: Shell-Closure.* Proc. Symp. on Freeform Fabrication, Dept. of Mech. Eng., Univ. of Texas at Austin, 1992.

[10] Guéziec, A.; Taubin, G.; Lazarus, F.; Horn, W.: *Cutting and Stitching: Efficient Conversion of a Non-Manifold Polygonal Surface to a Manifold.* IBM-RC-20935, IBM Research Div., Yorktown Heights, 1997.

[11] Carey, G.F.: *Computational Grids: Generation, Adaption, and Solution Strategies.* Taylor & Francis, 1997.

[12] Thompson, J.F.; Soni, B.K.; Weatherill, N.P. (eds.): *Handbook of Grid Generation.* CRC Press, Boca Raton, 1999.

[13] Gordon, W.N.; Hall, C.A.: *Construction of Curvilinear Coordinate Systems and Application to Mesh Generation.* Int. J. Num. Methods in Engineering, 7 (1973), pp. 461-477.

[14] Gordon, W.J.; Thiel, L.C.: *Transfinite Mappings and Their Application to Grid Generation.* Numerical Grid Generation, Thompson J.F. (ed.), North-Holland, 1982, p. 171.

[15] Thompson, J.F.; Warsi, Z.U.A.; Mastin, C.W.: *Numerical Grid Generation: Foundations and Applications.* Elsevier Science, 1985; see also at: `http://www.erc.msstate.edu/education/gridbook`.

[16] Thompson, J.F.; Thames, F.C.; Mastin, C.W.: *Automatic Numerical Generation of Body-Fitted Curvilinear Coordinate System for Field Containing Any Number of Arbitrary Two-Dimensional Bodies.* J. Computational Physics, 15 (1974), pp. 299-319.

[17] Sorenson, R.L.: *A Computer Program to Generate Two-Dimensional Grids About Airfoils and Other Shapes by the Use of Poisson's Equation.* NASA TM-81198, 1980.

[18] Hsu, K.; Lee, S.L.: *A Numerical Technique for Two-Dimensional Grid Generation with Grid Control at All of the Boundaries.* J. Computational Physics, 96 (1991), pp. 451-469.

[19] Thomas, P.D.; Middlecoff, J.F.: *Direct Control of the Grid Point Distribution in Meshes Generated by Elliptic Equations.* AIAA Journal, 18 (1980), pp. 652-656.

[20] Sorenson, R.L.: *Three-Dimensional Elliptic Grid Generation About Fighter Aircraft for Zonal Finite Difference Computations.* AIAA Paper 86-0429, 1986.

[21] Thompson, J.F.: *A General Three-Dimensional Elliptic Grid Generation System Based on a Composite Block Structure.* Comp. Meth. Appl. Mech. and Eng., 64 (1987), pp. 377-411.

[22] Sonar, T.: *Grid Generation Using Elliptic Partial Differential Equations.* DFVLR-FB 89-15, 1989.

[23] Starius, G.: *Constructing Orthogonal Curvilinear Meshes by Solving Initial Value Problems.* Numerische Mathematik, 28 (1977), pp. 25-48.

[24] Steger, J.L.; Chaussee, D.S.: *Generation of Body-Fitted Coordinates Using Hyperbolic Partial Differential Equations.* SIAM J. Sci. Stat. Comp., 1 (1980), pp. 431-437.

[25] Steger, J.L.; Rizh, Y.M.: *Generation of Three-Dimensional Body-Fitted Coordinates Using Hyperbolic Partial Differential Equations.* NASA TM-86753, 1985.

[26] Steger, J.L.: *Generation of Three-Dimensional Body-Fitted Grids by Solving Hyperbolic Partial Differential Equations.* NASA TM-101069, 1989.

[27] Dwyer, H.A.: *A Geometric Interpretation of Hyperbolic Grid Generation.* AIAA Paper 94-0315, 1994.

[28] Sethian, J.A.: *Curvature Flow and Entropy Conditions Applied to Grid Generation.* J. Computational Physics, 115 (1994), pp. 440-454.

[29] Jeng, Y.N.; Shu, Y.L.: *The Grid Combination Method for the Hyperbolic Grid Solver in Regions with Enclosed Boundaries.* AIAA Paper 95-0857, 1995.

[30] Jeng, Y.N.; Liou, Y.-C.: *Hyperbolic Equation Method of Grid Generation for Enclosed Regions.* AIAA Journal, 34 (1996), pp. 1293-1295.

[31] Mavriplis, D.J.: *An Advancing Front Delaunay Triangulation Algorithm Designed for Robustness.* ICASE Report No. 92-49, 1992; also AIAA Paper 93-0671, 1993; also J. Computational Physics, 117 (1995), pp. 90-101.

[32] Müller, J.D.; Roe, P.L.; Deconinck, H.: *A Frontal Approach for Internal Node Generation in Delaunay Triangulations.* Int. Journal of Numerical Methods in Fluids, 17 (1993), pp. 241-256.

[33] Müller, J.D.: *On Triangles and Flow.* PhD Thesis, The University of Michigan, 1996.

[34] Marcum, D.L.; Weatherill, N.P.: *Unstructured Grid Generation Using Iterative Point Insertion and Local Reconnection.* AIAA Journal, 33 (1995), pp. 1619-1625.

[35] Bern, M.; Eppstein, D.: *Quadrilateral Meshing by Circle Packing.* Proc. 16th Int. Meshing Roundtable, Park City, Utah, Oct. 13-15, 1997, pp. 7-19.

[36] Liu, J.; Tang, R.: *Ball-Packing Method: A New Approach for Quality Automatic Triangulation of Arbitrary Domains.* Proc. 16th Int. Meshing Roundtable, Park City, Utah, Oct. 13-15, 1997, pp. 85-96.

[37] Shimada, K.; et.al.: *Anisotropic Triangular Meshing of Parametric Surfaces via Close Packing of Ellipsoidal Bubbles.* Proc. 16th Int. Meshing Roundtable, Park City, Utah, Oct. 13-15, 1997, pp. 375-390.

[38] Mavriplis, D.J.: *Unstructured Mesh Generation and Adaptivity.* ICASE Report No. 95-26, 1995.

[39] Weatherill, N.P.; Hassan, O.; Morgan, K.; Peraire, J.; Peiro, J.; Marchant, M.J.: *Unstructured Grid Generation.* COSMASE: Short Course on Grid Generation, Automation and Parallel Utilisation, Lausanne, Switzerland, Sept. 23-27, 1996.

[40] Dirichlet, G.L.: *Über die Reduktion der positiven quadratischen Formen mit drei unbestimmten ganzen Zahlen.* Z. Reine Angew. Math., 40 (1850), pp. 209-227.

[41] Voronoj, G.: *Novelles applications de paramètres continus à la théorie des formes quadratiques.* Z. Reine Angew. Math., 134 (1908).

[42] Delaunay, B.: *Sur la sphère vide.* Bull. Acad. Science USSR, Class. Science Mat. Nat., 7 (1934), pp. 793-800.

[43] Holmes, D.G.; Snyder, D.D.: *The Generation of Unstructured Meshes Using Delaunay Triangulation.* Proc. 2nd Int. Conf. on Numerical Grid Generation in CFD, Pineridge, Swansea, Wales, UK, 1988, pp. 643-652.

[44] George, P.L.; Hecht, F.; Saltel, E.: *Fully Automatic Mesh Generator for 3-D Domains of any Shape.* Impact of Computing in Science and Eng., 2 (1990), pp. 187-218.

[45] Weatherill, N.P.; Hassan, O.; Marcum, D.L.: *Compressible Flowfield Solutions with Unstructured Grids Generated by Delaunay Triangulation.* AIAA Journal, 33 (1995), pp. 1196-1204.

[46] Rebay, S.: *Efficient Unstructured Mesh Generation by Means of Delaunay Triangulation and Bowyer-Watson Algorithm.* J. Computational Physics, 106 (1993), pp. 125-138.

[47] Baker, T.J.: *Triangulations, Mesh Generation and Point Placement Strategies.* Proc. Conf. Frontiers of Computational Fluid Dynamics 1994, Ithaca, New York, Nov. 1994.

[48] Green, P.J.; Sibson, R.: *Computing the Dirichlet Tessellation in the Plane.* Computer Journal, 21 (1977), pp. 168-173.

[49] Bowyer, A.: *Computing Dirichlet Tessellations.* Computer Journal, 24 (1981), pp. 162-166.

[50] Watson, D.F.: *Computing the n-Dimensional Delaunay Tessellation with Application to Voronoj Polytopes.* Computer Journal, 24 (1981), pp. 167-172.

[51] Ruppert, J.: *A Delaunay Refinement Algorithm for Quality 2-Dimensional Mesh Generation.* J. Algorithms, 18 (1995), pp. 548-585.

[52] Shewchuk, J.R.: *Triangle: Engineering a 2D Quality Mesh Generator and Delaunay Triangulator.* Proc. 1st Workshop Appl. Computational Geometry, Philadelphia, USA, May 1996, pp. 124-133.

[53] Finkel, R.A.; Bentley, J.L.: *Quad Trees, A Data Structure for Retrieval of Composite Keys.* Acta Informatica, 4 (1974).

[54] Samet, H.: *The Quadtree and Related Hierarchical Data Structures.* Computing Surveys, 16 (1984), pp. 188-260.

[55] Samet, H.: *The Design and Analysis of Spatial Data Structures.* Addison-Wesley, 1990.

[56] Yerry, M.A.; Shephard, M.S.: *A Modified Quadtree Approach to Finite Element Mesh Generation.* IEEE Comput. Graph. and Appl., 3 (1983), pp. 39-46.

[57] Yerry, M.A.; Shephard, M.S.: *Automatic Three-Dimensional Mesh Generation by the Modified-Octree Technique.* Int. J. Numer. Meth. Eng., 20 (1984), pp. 1965-1990.

[58] Baker, T.J.: *Three Dimensional Mesh Generation by Triangulation of Arbitrary Point Sets.* AIAA Paper 87-1124, 1987.

[59] Chew, L.P.: *Constrained Delaunay Triangulations.* Algorithmica, 4 (1989), pp. 97-108.

[60] George, P.L.; Hecht, F.; Saltel, E.: *Automatic Mesh Generator with Specified Boundary.* Comp. Meth. Appl. Mech. and Eng., 33 (1991), pp. 975-995.

[61] Weatherill, N.P.; Hassan, O.; Marcum, D.L.: *Calculation of Steady Compressible Flowfields with the Finite-Element Method.* AIAA Paper 93-0341, 1993.

[62] Sharov, D.; Nakahashi, K.: *A Boundary Recovery Algorithm for Delaunay Tetrahedral Meshing.* Proc. 5th Int. Conf. Num. Grid Generation in Comput. Field Simul., Mississippi State Univ., 1996, pp. 229-238.

[63] Peraire, J.; Vahdati, M.; Morgan, K.; Zienkiewicz, O.C.: *Adaptive Remeshing for Compressible Flow Computations.* J. Computational Physics, 72 (1987), pp. 449-466.

[64] Peraire, J.; Peiro, J.; Formaggia, L.; Morgan, K.; Zienkiewicz, O.C.: *Finite Element Euler Computations in Three Dimensions.* Int. J. Numerical Methods in Eng., 26 (1988), pp. 2135-2159.

[65] Löhner, R.; Parikh, P.: *Three-Dimensional Grid Generation by the Advancing Front Method.* Int. J. of Numerical Methods in Fluids, 8i (1988), pp. 1135-1149; also AIAA Paper 88-0515, 1988.

[66] Löhner, R.: *Progress in Grid Generation via the Advancing Front Technique.* Engineering with Computers, 12 (1996), pp. 186-210.

[67] Pirzadeh, S.: *Structured Background Grids for Generation of Unstructured Grids by Advancing-Front Method.* AIAA Journal, 31 (1993), pp. 257-265.

[68] Pirzadeh, S.: *Unstructured Viscous Grid Generation by Advancing-Front Method.* NASA CR-191449, 1993.

[69] Barth, T.J.: *On Unstructured Grids and Solvers.* VKI Lecture Series 1990-03, 1990, pp. 1-65.

[70] Marcum, D.L.: *Advancing-Front/Local-Reconnection (AFLR) Unstructured Grid Generation.* CFD Review 1998, Part I. Hafez, M.; Oshima, K. (eds.), World Scientific Computing Co., 1998.

[71] Babuška, I.; Aziz, A.K.: *On the Angle Condition in the Finite-Element Method.* SIAM J. Numerical Analysis, 13 (1976), pp. 214-226.

[72] Baker, T.J.: *Discretization of the Navier-Stokes Equations and Mesh Induced Errors.* Proc. 5th Int. Conf. on Numerical Grid Generation in CFD, Mississippi State University, Mississippi, April 1996.

[73] Baker, T.J.: *Irregular Meshes and the Propagation of Solution Errors.* Proc. 15th Int. Conf. on Numerical Methods in Fluid Dynamics, Monterey, CA, June 1996.

[74] Mavriplis, D.J.: *Adaptive Mesh Generation for Viscous Flows Using Delaunay Triangulation.* J. Computational Physics, 90 (1990), pp. 271-291.

[75] Vallet, M.G.; Hecht, F.; Mantel, B.: *Anisotropic Control of Mesh Generation Based upon a Voronoj Type Method.* Proc. 3rd Int. Conf. Num. Grid Generation in Comput. Fluid Dynamics an Related Fields, Barcelona, Spain. Arcilla, A.S. et al. (eds.), North-Holland, 1991, pp. 93-103.

[76] Irmisch, S.; Schwolow, R.: *Erzeugung unstrukturierter Dreiecknetze mittels der Delaunay-Triangulierung.* ZFW, 18 (1994), Springer Verlag, pp. 361-368.

[77] Pirzadeh, S.: *Unstructured Viscous Grid Generation by the Advancing-Layers Method.* AIAA Journal, 32 (1994), pp. 1735-1737; also AIAA Paper 93-3453, 1993.

[78] Pirzadeh, S.: *Viscous Unstructured Tree-Dimensional Grids by the Advancing-Layers Method.* AIAA Paper 94-0417, 1994.

[79] Hassan, O.; Probert, E.J.; Weatherill, N.P.; Marchant, M.J.; Morgan, K.; Marcum, D.L.: *The Numerical Simulation of Viscous Transonic Flow Using Unstructured Grids.* AIAA Paper 94-2346, 1994.

[80] Marchant, M.J.; Weatherill, N.P.: *Unstructured Grid Generation for Viscous Flow Simulations.* Proc. 4th Int. Conf. Num. Grid Generation in Comput. Fluid Dynamics an Related Fields, Swansea, UK. Weatherill, N.P. et al. (eds.), Pineridge Press, 1994, pp. 151-162.

[81] Piegl, L.; Tiller, W.: *The NURBS Book.* 2nd edition, Springer, 1997.

[82] Whitmire, J.B.; Dollar, T.W.: *Prismatic Grid Generation for NURBS Bounded Triangulations.* AIAA Paper 98-0219, 1998.

[83] Kallinderis, Y.; Ward, S.: *Prismatic Grid Generation with an Efficient Algebraic Method for Aircraft Configurations.* AIAA Paper 92-2721, 1992.

[84] Kallinderis, Y.; Ward, S.: *Prismatic Grid Generation for 3-D Complex Geometries.* AIAA Journal, 31 (1993), pp. 1850-1856.

[85] Kallinderis, Y.; Khawaja, A.; McMorris, H.: *Hybrid Prismatic/Tetrahedral Grid Generation for Viscous Flows Around Complex Geometries.* AIAA Journal, 34 (1996), pp. 291-298.

[86] Sharov, D.; Nakahashi, K.: *Hybrid Prismatic/Tetrahedral Grid Generation for Viscous Flow Applications.* AIAA Paper 96-2000, 1996; also AIAA Journal, 36 (1998), pp. 157-162.

[87] Connell, S.D.; Braaten, M.E.: *Semistructured Mesh Generation for Three-Dimensional Navier-Stokes Calculations.* AIAA Journal, 33 (1995), pp. 1017-1024.

[88] Dompierre, J.; Labbé, P.; Vallet, M.-G.; Camarero, R.: *How to Subdivide Pyramids, Prisms and Hexahedra into Tetrahedra.* Report CERCA R99-78, 1999; also 8th Int. Meshing Roundtable, Lake Tahoe, CA, 1999.

[89] Müller, J.D.: *Quality Estimates and Stretched Meshes Based on Delaunay Triangulations.* AIAA Journal, 32 (1994), pp. 2372-2379.

[90] Marcum, D.L.: *Generation of Unstructured Grids for Viscous Flow Applications.* AIAA Paper 95-0212, 1995.

[91] Marcum, D.L.; Gaither, J.A.: *Mixed Element Type Unstructured Grid Generation for Viscous Flow Applications.* AIAA Paper 99-3252, 1999.

[92] McMorris, H.; Kallinderis, Y.: *A Combined Octree-Advancing Front Method for Tetrahedra and Anisotropic Surface Meshes.* AIAA Paper 96-2442, 1996.

[93] McMorris, H.; Kallinderis, Y.: *Octree-Advancing Front Method for Generation of Unstructured Surface and Volume Meshes.* AIAA Journal, 35 (1997), pp. 976-984.

[94] Chen, H.; Bishop, J.: *Delaunay Triangulation for Curved Surfaces.* Proc. 6th Internat. Meshing Roundtable '97, October 13-15, 1997, Park City, Utah, USA, pp. 115-127.

[95] Nakahashi, K.: *FDM-FEM Zonal Approach for Computations of Compressible Viscous Flows.* Lecture Notes in Physics, 264 (1986), Springer Verlag, pp. 494-498.

[96] Nakahashi, K.; Obayashi, S.: *FDM-FEM Zonal Approach for Viscous Flow Computations over Multiple Bodies.* AIAA Paper 87-0604, 1987.

[97] Nakahashi, K.; Obayashi, S.: *Viscous Flow Computations Using a Composite Grid.* AIAA Paper 87-1128, 1987.

[98] Weatherill, N.P.: *Mixed Structured-Unstructured Meshes for Aerodynamic Flow Simulations.* Aeronautical Journal, 94 (1990), pp. 111-123.

[99] Shaw, J.A.; Georgala, J.M.; Peace, A.J.; Childs, P.N.: *The Construction, Application and Interpretation of Three-Dimensional Hybrid Meshes.* Proc. 3rd Int. Conf. Num. Grid Generation in Comput. Fluid Dynamics an Related Fields, Barcelona, Spain. Arcilla, A.S. et al. (eds.), North-Holland, 1991.

[100] Shaw, J.A.; Peace, A.J.: *Simulating Three-Dimensional Aeronautical Flows on Mixed Block-Structured/Semi-Unstructured/Unstructured Grids.* Technical Memorandum 428, Aircraft Research Association, Bedford, 1998.

[101] Pandya, S.A.; Hafez, M.M.: *A Semi-Implicit Finite-Volume Scheme for the Solution of Euler Equations on 3-D Prismatic Grids.* AIAA Paper 93-3431, 1993.

[102] Parthasarathy, V.; Kallinderis, Y.; Nakajima, K.: *Hybrid Adaption Method and Directional Viscous Multigrid with Prismatic Tetrahedral Meshes.* AIAA Paper 95-0670, 1995.

[103] Parthasarathy, V.; Kallinderis, Y.: *Directional Viscous Multigrid Using Adaptive Prismatic Meshes.* AIAA Journal, 33 (1995), pp. 69-78.

[104] Kao, K.H.; Liou, M.S.: *Direct Replacement of Arbitrary Grid-Overlapping by Nonstructured Grid.* AIAA Paper 95-0346, 1995.

[105] Nakahashi, K.: *Computations of Three-Dimensional Navier-Stokes Equations by Using a Prismatic Mesh.* Proc. 6th NAL Symposium on Aircraft Computational Aerodynamics, Tokyo, Japan, June 1988.

[106] Nakahashi, K.: *A Finite-Element Method on Prismatic Elements for the Three-Dimensional Navier-Stokes Equations.* Lecture Notes in Physics, 323 (1989), Springer Verlag, pp. 434-438.

[107] Nakahashi, K.: *Marching Grid Generation for External Viscous Flow Problems.* Trans. Japan Society of Aerospace Sciences, 35 (1992), pp. 88-102.

[108] Kallinderis, Y.: *A New Finite-Volume Navier-Stokes Scheme on Three-Dimensional Semi-Structured Prismatic Elements.* AIAA Paper 92-2697, 1992.

[109] Kallinderis, Y.: *Adaptive Hybrid Prismatic/Tetrahedral Grids.* Int. J. Numerical Methods in Fluids, 20 (1995), pp. 1023-1037.

[110] Melton, J.E.; Pandya, S.A; Steger, J.L.: *3-D Euler Flow Solutions Using Unstructured Cartesian and Prismatic Grids.* AIAA Paper 93-0331, 1993.

[111] Karman, S.L.: *SPLITFLOW: A 3-D Unstructured Cartesian/Prismatic Grid CFD Code for Complex Geometries.* AIAA Paper 95-0343, 1995.

[112] Smith, R.J.; Leschziner, M.A.: *Automatic Grid Generation for Complex Geometries.* Aeronautical Journal, 100 (1996), pp. 7-14.

[113] Wang, Z.J.; Hufford, G.S.; Przekwas, A.J.: *Adaptive Cartesian/Adaptive Prism (ACAP) Grid Generation for Complex Geometries.* AIAA Paper 97-0860, 1997.

[114] Delanaye, M.; Aftosmis, M.; Berger, M.J.; Liu, Y.; Pulliam, T.H.: *Automatic Hybrid-Cartesian Grid Generation for High-Reynolds Number Flows around Complex Geometries.* AIAA Paper 99-0777, 1999.

[115] Rai, M.M.: *A Conservative Treatment of Zonal Boundaries for Euler Equation Calculations.* J. Computational Physics, 62 (1986), pp. 472-503.

[116] Parthasarathy, V.N.; Graichen, C.M.; Hathaway, A.F.: *A Comparison of Tetrahedron Quality Measures.* Finite Elements in Analysis and Design, 15 (1993), pp. 255-261.

[117] Lawson, C.L.: *Software for C^1 Surface Interpolation.* Mathematical Software III, Rice, J.R. (ed.), Academic Press, 1977.

[118] Lawson, C.L.: *Properties of n-Dimensional Triangulations.* Computer Aided Geometric Design, 3 (1986), pp. 231-246.

Chapter 12

Description of the Source Codes

The accompanying CD-ROM contains the source codes of several 1-D and 2-D flow solvers and grid generators. Provided are also input datasets and grids for various 1-D and 2-D examples of flow cases. Furthermore, there are two programs for the von Neumann stability analysis of explicit and implicit time-stepping schemes. In total, the CD-ROM comprises more than 17 MBytes of source codes and data.

The aim of the software is to demonstrate how to translate the theoretical principles of the computational fluid dynamics, which were presented in the previous chapters, into a computer code. The programs should be conceived as a basis for further experimentation and enhancements. The source codes are provided under the terms of the GNU General Public License. See the file `LICENSE` in one of the directories for more details.

The source codes are written in standard FORTRAN-77 language with the exception of REAL*8 statements and few inline comments (! ...). However, this presents in general no problem to newer FORTRAN-77 or even FORTRAN-90 compilers. The programs do not contain any system calls or references to external libraries. The source codes are kept as simple as possible, but still flexible enough. No attempts were made to optimise for the execution speed or the memory. Where applicable, vectors of major variables (conservative variables, coordinates, metrics, residuals, etc.) are dimensioned in the main program `main.f` and then passed to the subroutines via parameter lists. The same holds also for the "dummy" vector `dum`, which is employed to store temporary variables. This means that the dimensions of the problem are to be set only in the main program, which has then to be recompiled. With a few exceptions, the source codes are organised in such a way that there is one subroutine or function in a file.

All grid, solution and convergence files are stored in plain ASCII format. The convergence and the solution files are written out in a form suitable for

visualisation with Tecplot[1]. However, the convergence files as well as the 1-D solution files can also be viewed using gnuplot[2], if the header lines are removed. The format of the convergence and solution files is always the same – there is one column for each variable. The names of the variables are given in the header. In the case of the solution files generated by the unstructured flow solver, the indices of the nodes of each element are stored after the coordinates and flow quantities. The first column represents node 1, the second one node 2, etc. The number of nodes is stored in the last line of the header (starting with ZONE) after "N=", the number of elements after "E=".

The directory structure on the CD-ROM consists of two highest-level directories: `dos` and `unix`. Both contain the same data, but in the first case formatted for DOS or Windows, and in the second case for Unix (also Linux). The CD-ROM itself is formatted according to the ISO-9660 file system with Rock Ridge and Joliet extensions. The format is fully compatible to Unix, Linux, Windows, and MS-DOS operating systems.

Within either the main `dos` or `unix` directory, you will find the following subdirectories:

- `analysis` – von Neumann stability analysis of 1-D model equations
- `grid1d` – 1-D grid generation (Laval nozzle)
- `grid2ds` – 2-D structured grid generation for external and internal flows
- `grid2du` – conversion of 2-D structured into unstructured triangular grids
- `struct1d` – solution of quasi 1-D Euler equations (nozzle flow)
- `struct2d` – solution of 2-D Euler equations on structured grids
- `unstr2d` – solution of 2-D Euler equations on unstructured triangular grids.

The contents of the above subdirectories is further explained in the sections below. In general, the subdirectory of each program contains a `README` file (always in the same place as the sources). It describes the particular files, how to compile and to run the application, and how the results are stored. Additionally, the `README` file explains the meaning of the main variables and of the input parameters. In the case of the flow solvers (i.e., `struct1d`, `struct2d` and `unstr2d`), the `README` file shows also the call tree. The subdirectories of the flow solvers contain a directory `run`, where you can find input files, grids and solutions for the various test cases.

---

[1] ©Amtec Engineering, Inc., PO Box 3633 Bellevue, WA 98009-3633, USA; http://www.amtec.com

[2] freely available for DOS, Unix, Linux and VMS platforms; http://www.cs.dartmouth.edu/gnuplot_info.html

The last remark concerns the convergence history. The convergence of all flow solvers provided here is measured as the 2-norm of the difference of the density variable from two consecutive time steps, i.e.,

$$\|\Delta\rho\|_2 = \sqrt{\sum_{I=1}^{N}(\rho_I^{n+1} - \rho_I^n)^2}\,, \tag{12.1}$$

where the summation is carried out over all N control volumes. It is convenient to normalise the convergence measure in Eq. (12.1) with its value from the first iteration. The programs store the convergence history directly in logarithmic scale.

12.1 Programs for Stability Analysis

The directory `dos\analysis` or `unix/analysis` contains two programs for the von Neumann stability analysis (Section 10.3) of linear 1-D model equations. The first program calculates the Fourier symbol and the magnitude of the amplification factor for the explicit multistage time-stepping scheme (Subsection 6.1.1) and the hybrid scheme (Subsection 6.1.2). The source code is provided in the subdirectory `mstage`. The second program analyses the damping properties of a general implicit scheme (Section 6.2). It is contained in the subdirectory `implicit`. Both programs can deal with either the 1-D convection or the 1-D convection-diffusion model equation (Subsections 10.3.2 and 10.3.3). The spatial discretisation can be conducted either by the central scheme with artificial dissipation, by the 1st-order upwind, or by the 2nd-order upwind scheme, respectively. The same schemes can be also applied to the implicit operator. In the case of the explicit scheme, the effect of the (central or upwind) implicit residual smoothing (Section 9.3) can be investigated as well.

12.2 Structured 1-D Grid Generator

The directory `dos\grid1d` or `unix/grid1d` contains a program for the generation of 1-D structured grids. Each grid node is associated with a certain area. The area distribution over the x-axis corresponds to the Laval nozzle. The area of the nozzle is calculated using the relation

$$\begin{aligned} A(x) &= 1 + \frac{1}{2}(A_1 - 1)\left\{1 + \cos\left(\frac{\pi x}{0.35}\right)\right\} && \text{for } 0 \le x \le 0.35 \\ A(x) &= 1 + \frac{1}{2}(A_2 - 1)\left\{1 - \cos\left[\frac{\pi(x - 0.35)}{0.65}\right]\right\} && \text{for } 0.35 < x \le 1\,, \end{aligned} \tag{12.2}$$

where the length of the nozzle is supposed to be equal to unity. Furthermore, in Eq. (12.2) A_1 represents the inlet area and A_2 the outlet area (see the sketch in Fig. 12.1). The area of the throat results from Eq. (12.2) to $A^*(x=0.35) = 1$.

The grid generated by this program serves as an input to the quasi 1-D Euler solver from Section 12.5.

12.3 Structured 2-D Grid Generators

The directory `dos\grid2ds` or `unix/grid2ds` contains three programs for the generation of 2-D structured grids for external and internal flows. The first program, which source code is provided in the subdirectory `cgrid`, generates a C-type grid (see Subsection 11.1.1) around an airfoil. An example can be seen in Fig. 11.6. The initial grid is generated algebraically by using the linear TFI method Eq. (11.5). Afterwards, elliptic PDE's (Subsection 11.1.3) are employed to produce boundary-orthogonal grid with specified wall spacing. The airfoil contour is approximated by a Bézier spline (see Refs. [1], [2] for an introduction to Bézier splines).

The second program is intended for the generation of an algebraic grid inside a channel with circular bump. The source code can be found in the subdirectory `channel`.

The next program, which is located in the subdirectory `hgrid`, generates an H-type grid (Subsection 11.1.1) in a cascade. An example of grid created by the program is presented in Fig. 11.8. The linear TFI technique (Eq. (11.5)) for algebraic grid generation is employed in this case. The contour of the blade is described again by a Bézier spline.

The programs `cgrid` and `hgrid` rely on the library provided in the subdirectory `srccom`. The library contains routines for spline interpolation, linear TFI, elliptic grid generation and grid stretching.

12.4 Structured to Unstructured Grid Converter

The directory `dos\grid2du` or `unix/grid2du` contains a program for the conversion of 2-D structured grids into unstructured triangular grids. The program divides each quadrilateral of the structured grid into two triangles by connecting the diagonal nodes with the shortest distance. The triangulated grids can be used as an input for the unstructured flow solver in Section 12.7.

12.5 Quasi 1-D Euler Solver

The directory `dos\struct1d` or `unix/struct1d` contains a program for the solution of quasi 1-D Euler equations. The equations govern the inviscid flow in a nozzle. They can be written in conservative, differential form as [3]

$$\frac{\partial \vec{W}}{\partial t} + \frac{\partial \vec{F}_c}{\partial x} = \vec{Q}\,. \tag{12.3}$$

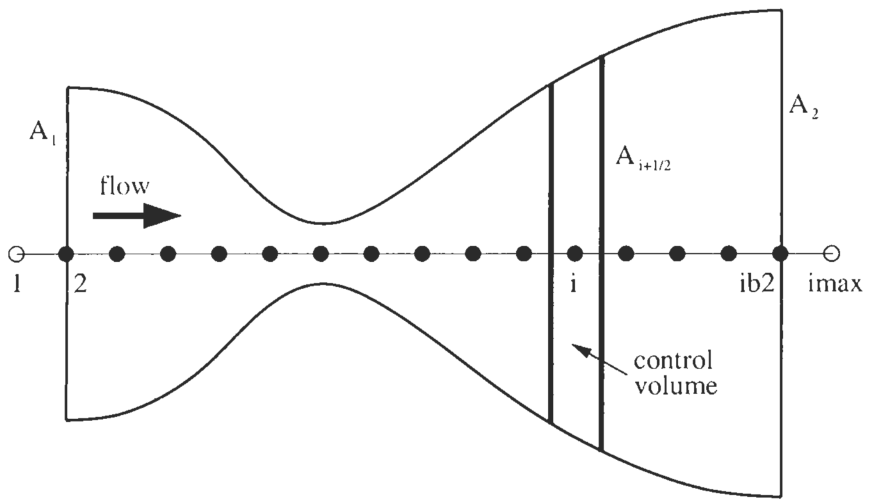


Figure 12.1: Grid and control volume for the 1-D Euler solver; points $i = 1$ and $i = imax$ are dummy points.

The vectors of conservative variables, convective fluxes, and the source term read

$$\vec{W} = \begin{bmatrix} \rho A \\ \rho u A \\ \rho E A \end{bmatrix}, \quad \vec{F}_c = \begin{bmatrix} \rho u A \\ (\rho u^2 + p) A \\ \rho H u A \end{bmatrix}, \quad \vec{Q} = \begin{bmatrix} 0 \\ p\, dA/dx \\ 0 \end{bmatrix}, \tag{12.4}$$

where A denotes the nozzle area. The total enthalpy H is given by the formula (2.12), and the pressure results according to Eq. (2.29) from

$$p = (\gamma - 1)\, \rho \left(E - \frac{u^2}{2} \right) . \tag{12.5}$$

The governing equations (12.3) are discretised on structured grid using the dual control-volume methodology. A sketch of the grid and of the control volume is displayed in Fig. 12.1. Central scheme with scalar artificial dissipation (Subsection 4.3.1), or optionally van Leer's flux-vector splitting scheme (Subsection 4.3.2) are employed for the spatial discretisation. The discretised equations are advanced in time using the explicit multistage scheme (Subsection 6.1.1 or 6.1.2). The program uses local time-stepping (Section 9.1) and the central implicit residual smoothing technique (Subsection 9.3.1) for convergence acceleration. The source code can be found in the subdirectory `src`.

The boundary conditions at the inlet and the outlet are implemented in characteristic variables as presented in Section 8.4. The concept of dummy points, described in Section 8.1, is utilised for this purpose (see Fig. 12.1).

12.6 Structured 2-D Euler Solver

The directory `dos\struct2d` or `unix/struct2d` contains a program for the solution of 2-D Euler equations on structured body-fitted grids. The spatial discretisation is based on the cell-centred finite-volume approach described in Subsection 4.2.1. It employs the central discretisation scheme with scalar artificial dissipation (Subsection 4.3.1). The governing equations are integrated in time using an explicit multistage scheme (Subsection 6.1.1 or 6.1.2), accelerated by local time-stepping (Section 9.1) and the central implicit residual smoothing (Subsection 9.3.1). The source code is provided in the subdirectory `src`.

The program utilises the concept of dummy cells (Section 8.1) for the treatment of the boundary conditions. Two layers of dummy cells are employed. A sketch of the grid in the computational domain is displayed in Fig. 12.2 (cf. Fig. 8.1). The program can deal only with single-block grids. However, it is very flexible with respect to the specification and the type of the boundary conditions. The following seven boundary types are implemented:

- coordinate cut,
- farfield (optionally with vortex correction),
- inflow,
- outflow,
- line periodic,
- solid wall, and
- symmetry.

The implementation of the above boundary conditions closely follows the discussion in Chapter 8.

The four boundaries of the computational space can be divided into an arbitrary number of segments. Each of the segments can be associated with a different boundary condition. This approach is quite similar to the description of block interfaces presented in Section 8.8. In fact, the program could be relatively easily extended to multiblock grids. Definitions of the segments are stored separately from the grid in the topology file (extension `.top`).

The definition of the face vectors `SI` and `SJ` (i.e., $\vec{n} \cdot \Delta S$) employed in the solver can be seen in Fig. 12.3. The face vectors are associated with the left and the bottom face of the control volume `VOL(I,J)` (corresponds to $\Omega_{I,J}$). They point outwards of the control volume. The face vectors of the remaining two sides of the control volume are obtained as `-SI(I+1,J)` and `-SJ(I,J+1)`. In this way, we have to store only two face vectors for each control volume.

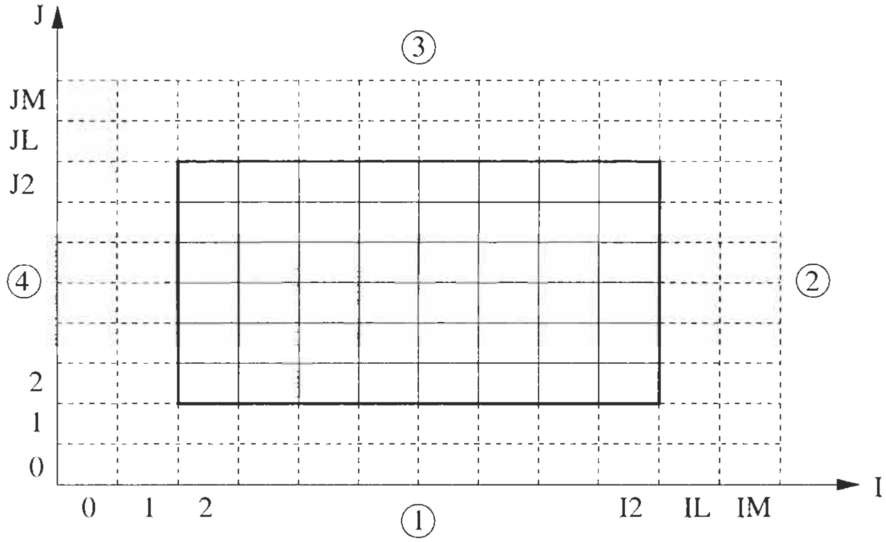


Figure 12.2: Grid topology in the computational space for the structured 2-D Euler solver. There are two layers of dummy cells at each boundary (dashed line). Numbers in circles denote the sides of the computational domain.

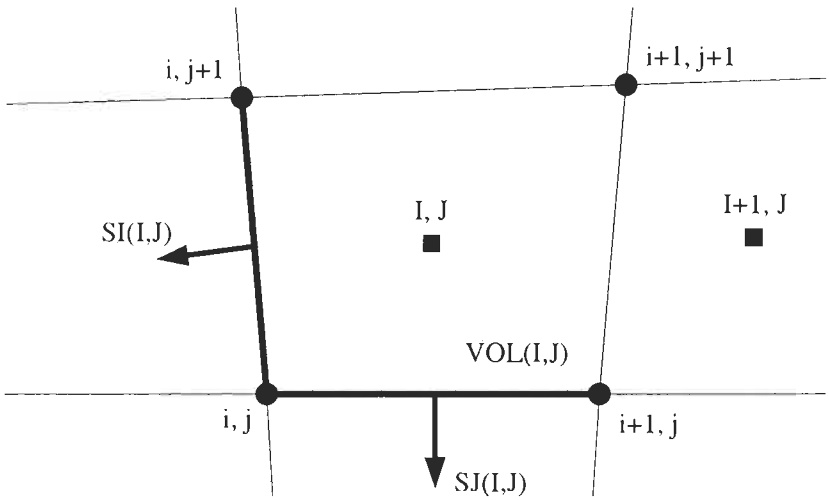


Figure 12.3: Control volume and face vectors for the structured 2-D Euler solver. Indices (I, J) denote the cell, indices (i, j) represent the grid node.

12.7 Unstructured 2-D Euler Solver

The directory `dos\unstr2d` or `unix/unstr2d` contains a program for the solution of 2-D Euler equations on unstructured triangular grids. The source code can be found in the subdirectory `src`. The median-dual cell-vertex scheme of Subsection 5.2.2 is utilised for the spatial discretisation. The data structure is edge oriented.

The convective fluxes are evaluated according to Roe's flux-difference splitting scheme (Subsection 4.3.3 and 5.3.2. The solution at the faces of the control volume is obtained by piecewise linear reconstruction of the left and right states as given by Eq. (5.38). The gradients of the flow variables are calculated with the Green-Gauss approach (Eqs. (5.45), (5.46)). The reconstructed solution is limited using Venkatakrishnan's limiter described in Subsection 5.3.5 (Eq. (5.63)).

The discretised governing equations are integrated in time using the explicit multistage scheme (Subsection 6.1.1 or 6.1.2), enhanced by local time-stepping (Section 9.1) and the central implicit residual smoothing (Subsection 9.3.2).

The same boundary conditions as described previously for the structured flow solver (except the coordinate cut) are implemented. The boundary conditions are imposed on boundary faces and not on nodes, in order to avoid any ambiguities (see the explanation at the beginning of Section 5.2). In principle, each boundary face can have a different boundary condition. The implementation of the farfield, inlet and outlet boundary conditions employs the concept of dummy nodes (one layer). All logical data related to the boundary conditions is stored together with the grid in one file (extension `.ugr`).

Bibliography

[1] Farin, G.E.: *Curves and Surfaces for Computer Aided Geometric Design – A Practical Guide.* 3rd ed., Academic Press, Boston, 1993.

[2] Hoschek, J., Lasser, D.: *Fundamentals of Computer Aided Geometric Design.* Wellesley, A.K. Peters Ltd., MA, 1993.

[3] Shapiro, A.H.: *The Dynamics and Thermodynamics of Compressible Fluid Flow.* Ronald Press, New York, 1953.

Appendix A

A.1 Governing Equations in Differential Form

Supposed the convective and viscous fluxes are continuous, the governing equations (2.19) can be transformed from integral to differential form by first applying Gauss's theorem. This leads to

$$\frac{\partial}{\partial t}\int_{\Omega} \vec{W}\, d\Omega + \int_{\Omega} \vec{\nabla} \cdot \left(\overline{\overline{F}}_c - \overline{\overline{F}}_v\right) d\Omega = \int_{\Omega} \vec{Q}\, d\Omega\,, \tag{A.1}$$

where $\overline{\overline{F}}_c$ and $\overline{\overline{F}}_v$ denote the tensors of convective and viscous fluxes, respectively. Equation (A.1) can then be written for an arbitrary control volume Ω in the differential form

$$\frac{\partial \vec{W}}{\partial t} + \vec{\nabla} \cdot \left(\overline{\overline{F}}_c - \overline{\overline{F}}_v\right) = \vec{Q}\,. \tag{A.2}$$

In order to account for arbitrary body-fitted grids, we introduce a mapping function between the Cartesian (x, y, z) and a curvilinear (ξ, η, ζ) coordinate system (cf. Fig. 3.2)

$$\begin{aligned} \xi &= \xi(x, y, z, t) \\ \eta &= \eta(x, y, z, t) \\ \zeta &= \zeta(x, y, z, t)\,. \end{aligned} \tag{A.3}$$

With this, the differential form of the three-dimensional Navier-Stokes equations (A.2) transforms into

$$\frac{\partial \vec{W}^*}{\partial t} + \frac{\partial \vec{F}_{c,1}}{\partial \xi} + \frac{\partial \vec{F}_{c,2}}{\partial \eta} + \frac{\partial \vec{F}_{c,3}}{\partial \zeta} = \frac{\partial \vec{F}_{v,1}}{\partial \xi} + \frac{\partial \vec{F}_{v,2}}{\partial \eta} + \frac{\partial \vec{F}_{v,3}}{\partial \zeta} + \vec{Q}^*\,. \tag{A.4}$$

The vector of the conservative variables is now given by

$$\vec{W}^* = J^{-1}\begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ \rho E \end{bmatrix}. \tag{A.5}$$

The vectors of the convective fluxes follow from the relationships

$$\vec{F}_{c,1} = J^{-1}\begin{bmatrix} \rho V_1 \\ \rho V_1 u + \xi_x p \\ \rho V_1 v + \xi_y p \\ \rho V_1 w + \xi_z p \\ \rho V_1 H \end{bmatrix}, \quad \vec{F}_{c,2} = J^{-1}\begin{bmatrix} \rho V_2 \\ \rho V_2 u + \eta_x p \\ \rho V_2 v + \eta_y p \\ \rho V_2 w + \eta_z p \\ \rho V_2 H \end{bmatrix},$$
$$\vec{F}_{c,3} = J^{-1}\begin{bmatrix} \rho V_3 \\ \rho V_3 u + \zeta_x p \\ \rho V_3 v + \zeta_y p \\ \rho V_3 w + \zeta_z p \\ \rho V_3 H \end{bmatrix}, \tag{A.6}$$

and the vectors of the viscous fluxes are defined as

$$\vec{F}_{v,1} = J^{-1}\begin{bmatrix} 0 \\ \xi_x\tau_{xx} + \xi_y\tau_{xy} + \xi_z\tau_{xz} \\ \xi_x\tau_{yx} + \xi_y\tau_{yy} + \xi_z\tau_{yz} \\ \xi_x\tau_{zx} + \xi_y\tau_{zy} + \xi_z\tau_{zz} \\ \xi_x\Theta_x + \xi_y\Theta_y + \xi_z\Theta_z \end{bmatrix},$$
$$\vec{F}_{v,2} = J^{-1}\begin{bmatrix} 0 \\ \eta_x\tau_{xx} + \eta_y\tau_{xy} + \eta_z\tau_{xz} \\ \eta_x\tau_{yx} + \eta_y\tau_{yy} + \eta_z\tau_{yz} \\ \eta_x\tau_{zx} + \eta_y\tau_{zy} + \eta_z\tau_{zz} \\ \eta_x\Theta_x + \eta_y\Theta_y + \eta_z\Theta_z \end{bmatrix}, \tag{A.7}$$
$$\vec{F}_{v,3} = J^{-1}\begin{bmatrix} 0 \\ \zeta_x\tau_{xx} + \zeta_y\tau_{xy} + \zeta_z\tau_{xz} \\ \zeta_x\tau_{yx} + \zeta_y\tau_{yy} + \zeta_z\tau_{yz} \\ \zeta_x\tau_{zx} + \zeta_y\tau_{zy} + \zeta_z\tau_{zz} \\ \zeta_x\Theta_x + \zeta_y\Theta_y + \zeta_z\Theta_z \end{bmatrix}.$$

Finally, the source term transforms to

$$\vec{Q}^* = J^{-1}\begin{bmatrix} 0 \\ \rho f_{e,x} \\ \rho f_{e,y} \\ \rho f_{e,z} \\ \rho\vec{f}_e\cdot\vec{v} + \dot{q}_h \end{bmatrix}. \tag{A.8}$$

where $\vec{f}_e$ represents the vector of the body forces.

The contravariant velocities in the ξ, η, ζ coordinate directions are given by the formulae

$$\begin{aligned} V_1 &= \xi_x u + \xi_y v + \xi_z w \\ V_2 &= \eta_x u + \eta_y v + \eta_z w \\ V_3 &= \zeta_x u + \zeta_y v + \zeta_z w\,. \end{aligned} \tag{A.9}$$

The components of the viscous stress tensor and the thermal fluxes read

$$
\begin{aligned}
\tau_{xx} &= 2\mu\frac{\partial u}{\partial x} + \lambda\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z}\right)\\
\tau_{yy} &= 2\mu\frac{\partial v}{\partial y} + \lambda\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z}\right)\\
\tau_{zz} &= 2\mu\frac{\partial w}{\partial z} + \lambda\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z}\right)\\
\tau_{xy} &= \tau_{yx} = \mu\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right)\\
\tau_{xz} &= \tau_{zx} = \mu\left(\frac{\partial u}{\partial z} + \frac{\partial w}{\partial x}\right)\\
\tau_{yz} &= \tau_{zy} = \mu\left(\frac{\partial v}{\partial z} + \frac{\partial w}{\partial y}\right)\\
\Theta_x &= u\tau_{xx} + v\tau_{xy} + w\tau_{xz} + k\frac{\partial T}{\partial x}\\
\Theta_y &= u\tau_{yx} + v\tau_{yy} + w\tau_{yz} + k\frac{\partial T}{\partial y}\\
\Theta_z &= u\tau_{zx} + v\tau_{zy} + w\tau_{zz} + k\frac{\partial T}{\partial z}\,.
\end{aligned}
\tag{A.10}
$$

In the above Eq. (A.10), k stands for the thermal conductivity coefficient, μ for the coefficient of dynamic viscosity, and λ denotes the second viscosity coefficient ($\lambda = -(2/3)\mu$ according to Stokes's hypothesis). The diffusive thermal fluxes can also be written in the following modified form

$$
\begin{aligned}
k\,\frac{\partial T}{\partial x} &= \frac{\mu}{(\gamma-1)Pr}\,\frac{\partial c^2}{\partial x}\\
k\,\frac{\partial T}{\partial y} &= \frac{\mu}{(\gamma-1)Pr}\,\frac{\partial c^2}{\partial y}\\
k\,\frac{\partial T}{\partial z} &= \frac{\mu}{(\gamma-1)Pr}\,\frac{\partial c^2}{\partial z}
\end{aligned}
\tag{A.11}
$$

with c representing the speed of sound and Pr the Prandtl number. The derivatives of the velocity components u, v, w and the temperature T in Cartesian coordinates x, y, z can be expressed with the aid of the chain rule as derivatives of ξ, η and ζ. For example,

$$
\begin{aligned}
\frac{\partial u}{\partial x} &= \xi_x\frac{\partial u}{\partial \xi} + \eta_x\frac{\partial u}{\partial \eta} + \zeta_x\frac{\partial u}{\partial \zeta}\\
\frac{\partial u}{\partial y} &= \xi_y\frac{\partial u}{\partial \xi} + \eta_y\frac{\partial u}{\partial \eta} + \zeta_y\frac{\partial u}{\partial \zeta}\,, \quad \text{etc.}
\end{aligned}
\tag{A.12}
$$

The inverse of the determinant of the coordinate transformation Jacobian $\partial(\xi,\eta,\zeta)/\partial(x,y,z)$ is defined as

$$
\begin{aligned}
J^{-1} = {} & \frac{\partial x}{\partial \xi}\frac{\partial y}{\partial \eta}\frac{\partial z}{\partial \zeta} + \frac{\partial x}{\partial \eta}\frac{\partial y}{\partial \zeta}\frac{\partial z}{\partial \xi} + \frac{\partial x}{\partial \zeta}\frac{\partial y}{\partial \xi}\frac{\partial z}{\partial \eta} - \\
& \frac{\partial x}{\partial \xi}\frac{\partial y}{\partial \zeta}\frac{\partial z}{\partial \eta} - \frac{\partial x}{\partial \eta}\frac{\partial y}{\partial \xi}\frac{\partial z}{\partial \zeta} - \frac{\partial x}{\partial \zeta}\frac{\partial y}{\partial \eta}\frac{\partial z}{\partial \xi}\,.
\end{aligned}
\tag{A.13}
$$

Finally, the metric terms are specified through the relations

$$
\begin{aligned}
\xi_x &= J\left(\frac{\partial y}{\partial \eta}\frac{\partial z}{\partial \zeta} - \frac{\partial y}{\partial \zeta}\frac{\partial z}{\partial \eta}\right)\\
\xi_y &= J\left(\frac{\partial z}{\partial \eta}\frac{\partial x}{\partial \zeta} - \frac{\partial z}{\partial \zeta}\frac{\partial x}{\partial \eta}\right)\\
\xi_z &= J\left(\frac{\partial x}{\partial \eta}\frac{\partial y}{\partial \zeta} - \frac{\partial x}{\partial \zeta}\frac{\partial y}{\partial \eta}\right)\\
\eta_x &= J\left(\frac{\partial y}{\partial \zeta}\frac{\partial z}{\partial \xi} - \frac{\partial y}{\partial \xi}\frac{\partial z}{\partial \zeta}\right)\\
\eta_y &= J\left(\frac{\partial z}{\partial \zeta}\frac{\partial x}{\partial \xi} - \frac{\partial z}{\partial \xi}\frac{\partial x}{\partial \zeta}\right)\\
\eta_z &= J\left(\frac{\partial x}{\partial \zeta}\frac{\partial y}{\partial \xi} - \frac{\partial x}{\partial \xi}\frac{\partial y}{\partial \zeta}\right)\\
\zeta_x &= J\left(\frac{\partial y}{\partial \xi}\frac{\partial z}{\partial \eta} - \frac{\partial z}{\partial \xi}\frac{\partial y}{\partial \eta}\right)\\
\zeta_y &= J\left(\frac{\partial z}{\partial \xi}\frac{\partial x}{\partial \eta} - \frac{\partial x}{\partial \xi}\frac{\partial z}{\partial \eta}\right)\\
\zeta_z &= J\left(\frac{\partial x}{\partial \xi}\frac{\partial y}{\partial \eta} - \frac{\partial y}{\partial \xi}\frac{\partial x}{\partial \eta}\right)\,.
\end{aligned}
\tag{A.14}
$$

It is important to see how the above metric terms from Eqs. (A.13), (A.14) are related to geometrical quantities like volume or face vectors. Let us for this purpose consider a 2-D structured grid, as it is sketched in Fig. A.1. We want further assume that the ξ-coordinate corresponds to the i-direction, and the η-coordinate to the j-direction, respectively. Then, the derivatives of the Cartesian coordinates with respect to the curvilinear coordinates in Eq. (A.14) can be approximated at point $(i,\ j)$ as

$$
\begin{aligned}
\left(\frac{\partial x}{\partial \xi}\right)_{i,j} &\approx \frac{\frac{1}{2}\left[(x_{i,j}+x_{i+1,j})-(x_{i,j}+x_{i-1,j})\right]}{(i+1/2)-(i-1/2)} = \frac{1}{2}\left(x_{i+1,j}-x_{i-1,j}\right)\\
\left(\frac{\partial y}{\partial \xi}\right)_{i,j} &\approx \frac{\frac{1}{2}\left[(y_{i,j}+y_{i+1,j})-(y_{i,j}+y_{i-1,j})\right]}{(i+1/2)-(i-1/2)} = \frac{1}{2}\left(y_{i+1,j}-y_{i-1,j}\right)
\end{aligned}
\tag{A.15}
$$

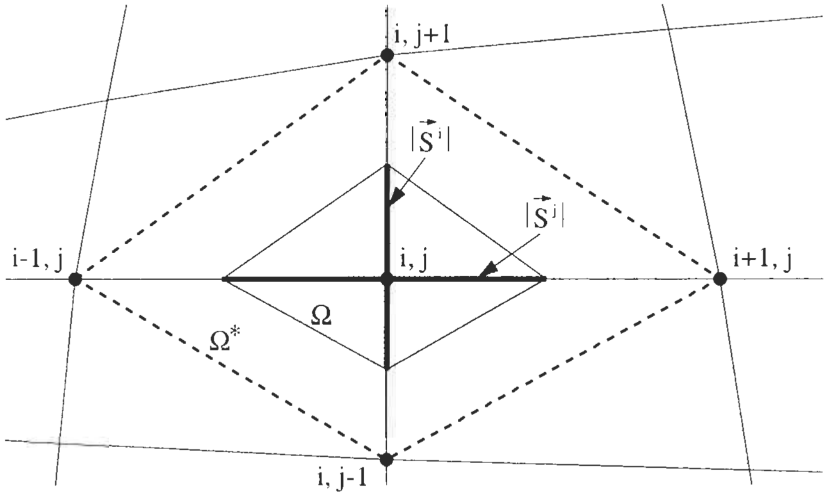


Figure A.1: Geometrical representation of the metric terms in two dimensions.

and

$$\left(\frac{\partial x}{\partial \eta}\right)_{i,j} \approx \frac{\frac{1}{2}\left[(x_{i,j}+x_{i,j+1})-(x_{i,j}+x_{i,j-1})\right]}{(j+1/2)-(j-1/2)} = \frac{1}{2}(x_{i,j+1}-x_{i,j-1})$$
$$\left(\frac{\partial y}{\partial \eta}\right)_{i,j} \approx \frac{\frac{1}{2}\left[(y_{i,j}+y_{i,j+1})-(y_{i,j}+y_{i,j-1})\right]}{(j+1/2)-(j-1/2)} = \frac{1}{2}(y_{i,j+1}-y_{i,j-1})\,. \tag{A.16}$$

Thus, using the above approximations of Eqs. (A.15) and (A.16), the formulae in Eq. (A.14) become

$$\begin{aligned}
\xi_x &= \quad J\,\frac{\partial y}{\partial \eta} \approx J\,S_x^I \\
\xi_y &= -\,J\,\frac{\partial x}{\partial \eta} \approx J\,S_y^I \\
\eta_x &= -\,J\,\frac{\partial y}{\partial \xi} \approx J\,S_x^J \\
\eta_y &= \quad J\,\frac{\partial x}{\partial \xi} \approx J\,S_y^J\,,
\end{aligned} \tag{A.17}$$

where $\vec{S}^I = [S_x^I, S_y^I]^T$ denotes the average face vector in i-direction, and $\vec{S}^J = (S_x^J, S_y^J)^T$ the vector in j-direction, respectively (cf. Fig. A.1). The relationships in Eq. (A.17) can also be expressed in the form

$$\vec{S}^I = J^{-1}\begin{bmatrix}\xi_x\\ \xi_y\end{bmatrix}, \quad \vec{S}^J = J^{-1}\begin{bmatrix}\eta_x\\ \eta_y\end{bmatrix}. \tag{A.18}$$

From the above we can see that the metric terms correspond to the components of face vectors in the finite volume scheme, divided by the determinant of the coordinate transformation Jacobian J.

The question is now, what is the geometrical interpretation of J^{-1}? Using Eq. (A.13) which reads in two dimensions as

$$J^{-1} = \frac{\partial x}{\partial \xi}\frac{\partial y}{\partial \eta} - \frac{\partial x}{\partial \eta}\frac{\partial y}{\partial \xi}, \tag{A.19}$$

and inserting the expressions Eq. (A.17) for the metric terms into Eq. (A.19), it can be shown that

$$J^{-1} = 2\Omega = \frac{1}{2}\Omega^* . \tag{A.20}$$

The volumes Ω and Ω^* (with constant depth b=1) are displayed in Fig. A.1. They are defined by the points (i+1/2, j), (i, j+1/2), (i-1/2, j), (i, j-1/2) in the case of Ω, and by the grid nodes (i, j-1), (i+1, j), (i, j+1), (i-1, j) in the case of Ω^*, respectively. It should be noted that on rectangular grid J^{-1} equals to the dual control volume which was presented in Subsection 4.2.3.

The same derivations for the metric terms as above can be conducted in three dimensions. The results are shown in Fig. A.2. The inverse of the determinant of the transformation Jacobian (J^{-1}) equals to twice the volume defined by the grid nodes (i, j, k-1/2), (i+1/2, j, k), (i, j, k+1/2), (i-1/2, j, k), (i, j-1/2, k), and (i, j+1/2, k). On rectangular grid this again corresponds exactly to the dual control volume of Subsection 4.2.3.

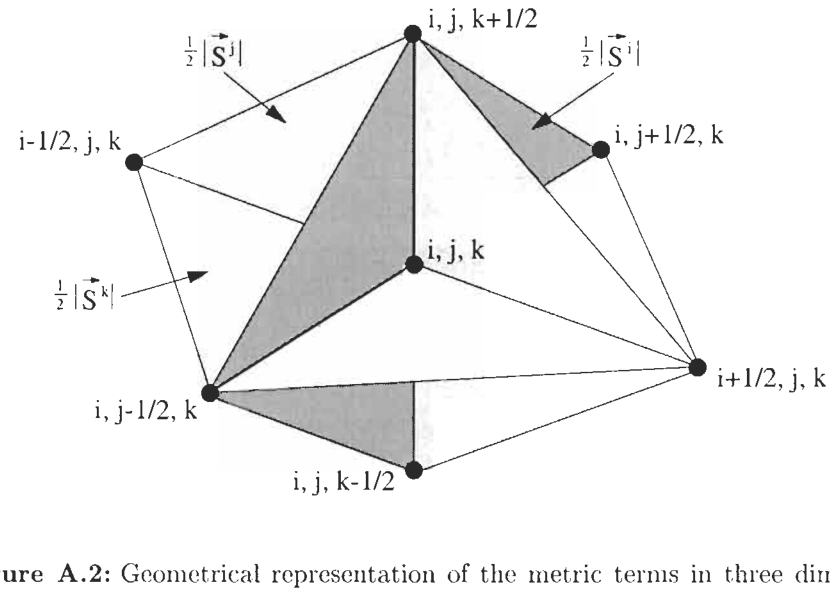


Figure A.2: Geometrical representation of the metric terms in three dimensions.

A.2 Mathematical Character of the Governing Equations

The mathematical character of the partial differential equations (PDE's) is best explained through the classical example of the quasi-linear second-order equation

$$a\,\frac{\partial^2 U}{\partial x^2} + b\,\frac{\partial^2 U}{\partial x \partial y} + c\,\frac{\partial^2 U}{\partial y^2} = d\,, \tag{A.21}$$

where U represents a general scalar function. The coefficients a, b, c and d may be non-linear functions of the coordinates, of U and of its first derivatives, but not of the second derivatives of U. Depending on the sign of the discriminant function $(b^2 - 4ac)$, three different classes of PDE's can be defined [1], [2]. Namely, if the discriminant is positive the equation is said to be ***hyperbolic***, whereas if $(b^2 - 4ac) < 0$ it becomes *elliptic*. Finally, if $(b^2 - 4ac)$ is zero the PDE is denoted as ***parabolic***.

The Navier-Stokes equations cannot be characterised in such an easy way. In fact, they are in general a mixture of all three classes, depending on the flow conditions and on the geometry of the problem. Nevertheless, it may be worthwhile to illustrate the physical behaviour of hyperbolic, parabolic and elliptic PDE's which must be considered for proper mathematical formulation of solution methods in fluid dynamics.

A.2.1 Hyperbolic Equations

If the Equation (A.21) is of hyperbolic type, it has two real characteristics. The situation is sketched in Fig. A.3). It is known from the theory that the information at point P influences only the region between the advancing characteristics. On the other hand, point P receives information only from the part of the domain between the characteristics AP and BP. Furthermore, the solution at point P depends only on that part of the boundary which is intercepted by and included within the two characteristic lines through point P, i.e., the interval AB. For example, point P obtains no information from point C, since P is not enclosed within the characteristics propagating from C. Therefore, we need to specify conditions at only one part of the boundary in order to determine the solution in a given region. Hence, the hyperbolic equations represent an *initial-value* problem.

In fluid dynamics, the following are examples of flows which are governed by hyperbolic partial differential equations:

1. Steady, inviscid supersonic flow:

   if the flow is two-dimensional, the behaviour is like that already discussed above. For a 3-D flow, there are characteristic surfaces in (x, y, z) space. Information at P influences the volume within the advancing (downstream) characteristic surface. The characteristic surface corresponds to the Mach cone in the case of linearised potential equation.

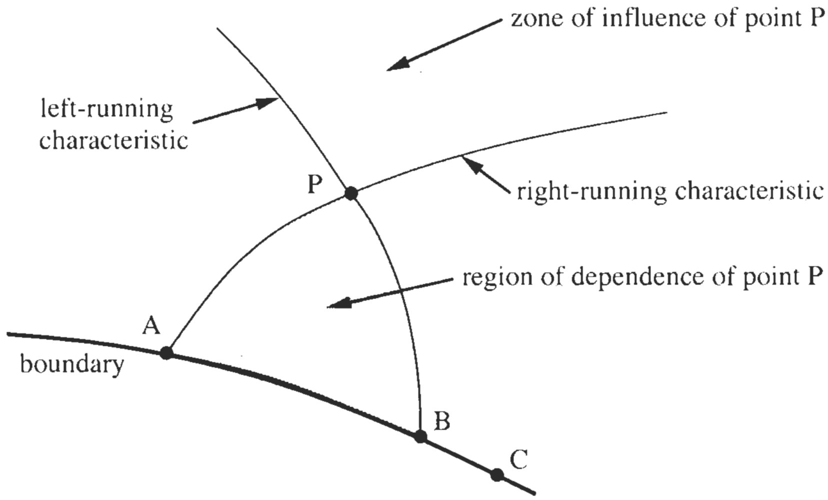


Figure A.3: Domain of dependence and influence of hyperbolic PDE with two characteristics per point [3].

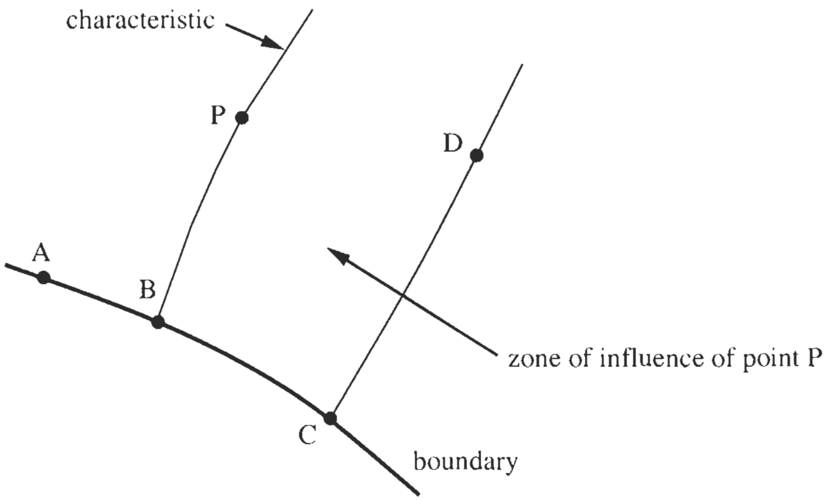


Figure A.4: Domain of influence of parabolic PDE [3].

2. Unsteady flow:

   the governing equations are in this case hyperbolic in **time**, no matter whether the flow is locally subsonic or supersonic. Of course, the equations can be of different type in **space**. However, because of the (partial) hyperbolic nature, it is necessary to specify an initial solution which is then advanced in time.

A.2.2 Parabolic Equations

In this case, Equation (A.21) has only one real characteristic. Fig. A.4 shows the characteristic together with the domain of influence. For parabolic equations, information at point P influences the entire region on one side of the characteristic (BP in the sketch) and hence also the points C and D. On the other hand, the solution at point A is completely independent of that at point P. Hence, the information is transfered in one direction only, like for hyperbolic PDE's. Furthermore, as we can see from Fig. A.4, for example the solution at point D depends on conditions along the whole characteristic BP as well as on those prescribed at the boundary segment BC. Thus, the parabolic equations represent a mixed *initial-* and *boundary-value* problem.

One particular form of the simplified governing equations, namely the so-called Parabolised Navier-Stokes (PNS) equations (see Subsection 2.4.3) exhibit parabolic-type behaviour. In this case, the viscous stress terms involving the derivatives with respect to the streamwise direction are ignored. The solution of the PNS equations is started from some prescribed (initial) data at the inlet boundary. The simulation then proceeds by marching downstream. Each streamwise station represents a plane (line in 2-D) which is perpendicular to the main flow direction and which is coupled in an explicit way to one or two upstream planes. In each plane, the equations have to be solved iteratively.

A further simplification of the Navier-Stokes equations for the case of high Reynolds numbers leads to the well-known boundary layer equations. These are also of parabolic type and they can be solved by a similar explicit space marching procedure.

A.2.3 Elliptic Equations

If the Equation (A.21) is of elliptic type, it has two complex characteristics. It is known from the theory that the information at some point P influences all other regions of the domain. On the other hand, the solution at point P depends on the surrounding domain. The situation is sketched in Fig. A.5. For elliptic equations, because point P has an effect on all points of the domain, then in turn the solution at point P is influenced by the entire closed boundary (A, B, C). Therefore, the elliptic PDE's represent a *boundary-value* problem, where the solution at point P must be carried out simultaneously with the solution at all other points in the domain. This is in strong contrast to the space or time marching procedures applied to parabolic and hyperbolic equations. In terms

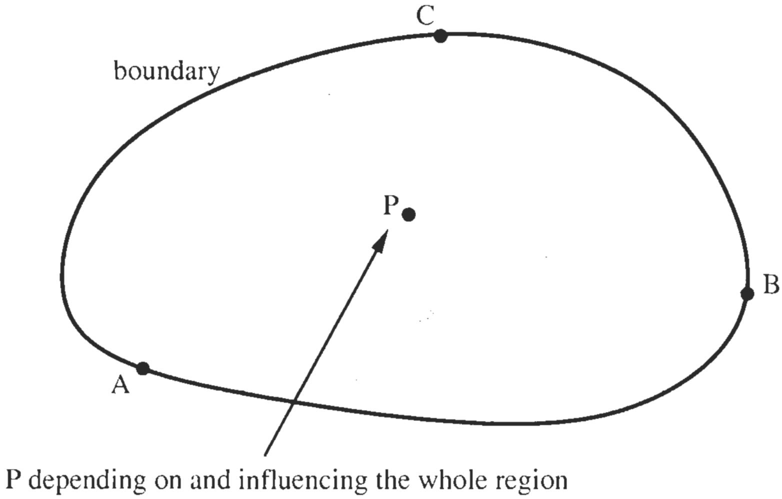


Figure A.5: Domain of influence and dependence of elliptic PDE [3].

of Fig. A.5, boundary conditions must be prescribed over the entire boundary (A, B, C). The conditions can take the following forms:

1. Specification of the dependent variables along the boundary. This type of boundary condition is called the *Dirichlet condition.*

2. Specification of the derivatives of the dependent variables along the boundary. This type of boundary condition is named the *Neumann condition.*

In fluid dynamics, for example, the steady subsonic/incompressible inviscid flow is governed by elliptic equations. Hence, for such flows physical boundary conditions must be applied to a surface which completely surrounds the flow field.

A.3 Navier-Stokes Equations in Rotating Frame of Reference

In some instances, for example in turbomachinery applications, for propellers or in geophysics, the computational domain is steadily rotating about some axis. In such a case, it is convenient to transform the Navier-Stokes equations into a *rotating frame of reference*. Let us consider the situation which is depicted in Fig. A.6. Here, a point P rotates with the constant angular velocity $\vec{\omega}$ around a fixed axis. For the sake of simplicity, we assume that the rotation axis coincides with the x-coordinate axis. Thus, the angular velocity has the components $\vec{\omega} = [\omega_1, 0, 0]^T$. The absolute velocity $\vec{v}_a$ results from the sum of the relative velocity $\vec{v}_r$ and the entrainment velocity $\vec{v}_e$, i.e.,

$$\vec{v}_a = \vec{v}_r + \vec{v}_e = \vec{v}_r + \vec{\omega} \times \vec{r}\,. \tag{A.22}$$

If we re-write the Navier-Stokes equations (2.19) in relative frame of reference, we have to account for the effects due to the *Coriolis force* as well as due to the *centrifugal force*. The Coriolis force per unit mass is defined as

$$\vec{f}_{Corr} = -2\,(\vec{\omega} \times \vec{v}_r)\,. \tag{A.23}$$

The centrifugal force per unit mass is given by

$$\vec{f}_{cen} = -\vec{\omega} \times (\vec{\omega} \times \vec{r}) = \omega_1^2\, \vec{r}_n\,, \tag{A.24}$$

where $\vec{r}_n$ denotes the position vector perpendicular to the rotation axis. In consequence, the source term $\vec{Q}$ (Eq. (2.25)) has to be extended by the sum of the Coriolis and the centrifugal forces for the momentum equations. Furthermore, the energy equation has to be modified because of the centrifugal force (the Coriolis force does not contribute to the energy balance). Only the continuity equation stays unchanged since the mass balance is invariant to system rotation. Hence, the Navier-Stokes equations formulated in relative frame of reference which rotates with constant angular velocity about the x-axis reads

$$\frac{\partial}{\partial t}\int_\Omega \vec{W}\, d\Omega + \oint_{\partial\Omega} (\vec{F}_c - \vec{F}_v)\, dS = \int_\Omega \vec{Q}\, d\Omega\,. \tag{A.25}$$

The vector of the conservative variables $\vec{W}$ consists of the following components

$$\vec{W} = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ \rho E \end{bmatrix}, \tag{A.26}$$

where ρ, u, v, w, E denote the density, the Cartesian velocity components in relative frame, and the relative total energy per unit mass, respectively. The relative total energy is given by

$$E = e + \frac{|\vec{v}_r|^2}{2} - \frac{|\vec{v}_e|^2}{2} = e + \frac{u^2 + v^2 + w^2}{2} - \frac{\omega_1^2\, |\vec{r}_n|^2}{2} \tag{A.27}$$

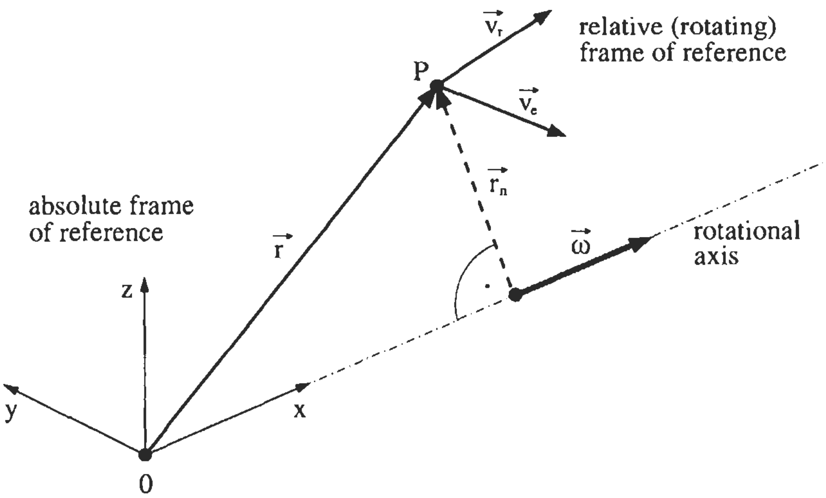


Figure A.6: Absolute and rotating frame of reference.

with $|\vec{r}_n|^2 = y^2 + z^2$. The vector of the convective fluxes $\vec{F}_c$ is defined as

$$\vec{F}_c = \begin{bmatrix} \rho V \\ \rho u V + n_x p \\ \rho v V + n_y p \\ \rho w V + n_z p \\ \rho I V \end{bmatrix}, \tag{A.28}$$

with p being the static pressure, and n_x, n_y, n_z representing the components of the outward pointing unit normal vector of the surface $\partial\Omega$, respectively. Furthermore,

$$\begin{aligned} I &= h + \frac{|\vec{v}_r|^2}{2} - \frac{|\vec{v}_e|^2}{2} = H - \frac{\omega_1^2\,|\vec{r}_n|^2}{2} \\ V &= n_x u + n_y v + n_z w\,, \end{aligned} \tag{A.29}$$

where I stands for the *rothalpy*, H for the relative total enthalpy, and V for the contravariant velocity, respectively. The rothalpy represents the total energy content in a steadily rotating frame of reference.

The vector of the viscous fluxes $\vec{F}_v$ takes the same form as presented in Eq. (2.23). Also the components of the viscous stress tensor remain formally as given by Eq. (2.15). However, the source term $\vec{Q}$ is extended by the Coriolis and the centrifugal force to

$$\vec{Q} = \begin{bmatrix} 0 \\ \rho f_{e,x} \\ \rho\,\omega_1(y\,\omega_1 + 2w) + \rho f_{e,y} \\ \rho\,\omega_1(z\,\omega_1 - 2v) + \rho f_{e,z} \\ \rho \vec{f}_e \cdot \vec{v}_r + \dot{q}_h \end{bmatrix} \tag{A.30}$$

with $\rho \vec{f_e}$ being the body force per unit volume (in addition to Coriolis and centrifugal forces), and $\dot{q}_h$ being the time rate of heat transfer per unit mass, respectively.

The governing equations (A.25) are closed using thermodynamic relations between the state variables. In the case of a perfect gas, the pressure is computed from the formula

$$p = (\gamma - 1)\rho \left[E - \frac{u^2 + v^2 + w^2 - \omega_1^2 \left| \vec{r}_n \right|^2}{2} \right], \qquad \text{(A.31)}$$

where γ denotes the ratio of specific heat coefficients. Additionally, viscosity and thermal conductivity coefficients have to be supplied as a function of the state of the fluid.

A.4 Navier-Stokes Equations Formulated for Moving Grids

In certain cases, where for instance the fluid-structure interaction is investigated or where the store separation is simulated, it is necessary to solve the governing equations on a moving and possibly also a deforming grid. The two most popular methodologies used to tackle such problems, are the *Arbitrary Lagrangian Eulerian* (ALE) formulation [4]-[6] and the *dynamic grids* [7]. Both approaches are closely related and lead to the same modified form of the governing equations which accounts for the relative motion of the grid with respect to the fluid.

Written in time-dependent integral form for a moving and/or deforming control volume Ω with a surface element dS, the Navier-Stokes equations (2.19) read

$$\frac{\partial}{\partial t}\int_{\Omega} \vec{W} d\Omega + \oint_{\partial\Omega} (\vec{F}_c^M - \vec{F}_v)\, dS = \int_{\Omega} \vec{Q}\, d\Omega . \tag{A.32}$$

The vector of the conservative variables $\vec{W}$ has the following components

$$\vec{W} = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ \rho E \end{bmatrix} , \tag{A.33}$$

where ρ, u, v, w, E denote the density, the Cartesian velocity components and the total energy per unit mass, respectively. The vector of the convective fluxes $\vec{F}_c^M$ becomes on dynamic grids

$$\vec{F}_c^M = \vec{F}_c - V_t \vec{W} \tag{A.34}$$

with $\vec{F}_c$ given by Eq. (2.21) and V_t being the contravariant velocity of the face of the control volume. Hence,

$$V_t = n_x \frac{\partial x}{\partial t} + n_y \frac{\partial y}{\partial t} + n_z \frac{\partial z}{\partial t} . \tag{A.35}$$

In Eq. (A.35), n_x, n_y, and n_z denote the components of the outward facing unit normal vector of the surface $\partial\Omega$. Using Eq. (2.21), the convective fluxes $\vec{F}_c^M$ can be written in the form

$$\vec{F}_c^M = \begin{bmatrix} \rho V_r \\ \rho u V_r + n_x p \\ \rho v V_r + n_y p \\ \rho w V_r + n_z p \\ \rho H V_r + V_t p \end{bmatrix} , \tag{A.36}$$

where H stands for the total enthalpy and p for the static pressure, respectively. Furthermore, V_r represents the contravariant velocity relative to the motion of the grid, i.e.,

$$V_r = n_x u + n_y v + n_z w - V_t = V - V_t . \tag{A.37}$$

The vector of the viscous fluxes $\vec{F}_v$ and the source term $\vec{Q}$ retain the same forms as already presented in Eq. (2.23) and Eq. (2.25), respectively. The same holds also for the components of the viscous stress tensor Eq. (2.15).

It was first pointed out by Thomas and Lombard [8] that besides the conservation of mass, momentum and energy, the so-called *Geometric Conservation Law* (GCL) must be satisfied in order to avoid errors induced by deformation of control volumes [7], [9]-[11]. The integral form of the GCL reads

$$\frac{\partial}{\partial t}\int_{\Omega} d\Omega - \oint_{\partial\Omega} V_t \, dS = 0 \, . \tag{A.38}$$

The GCL results from the requirement that the computation of the control volumes or of the grid velocities must be performed in such a way that the resulting numerical scheme preserves the state of a uniform flow, independently of the deformation of the grid [11]. It should be stressed that the GCL is automatically satisfied for such moving grids, where the shapes of the control volumes do not change in time.

It is essential that the GCL in Eq. (A.38) is temporally discretised using the same scheme as it is applied to the governing equations (A.32) in order to obtain a self-consistent solution method. Furthermore, the GCL has to be solved concurrently with the fluid equations. In fact, both requirements represent a condition for the spatial discretisation of Eq. (A.32). More details on the numerical implementation of the governing equations for moving and/or deforming grids and on the discretisation of the GCL can be found in the above cited references and also e.g. in [12]-[15]. A very detailed description of a scheme for unstructured grids was presented in [16].

A.5 Thin Shear Layer Approximation

In the case of high Reynolds numbers, the flow is influenced by the viscous stresses only in a narrow region around the body. It can then be assumed that only the gradients in the normal direction to the wall dominate, and the gradients in the other directions can be neglected [17], [18]. We speak then of the so-called *Thin Shear Layer* (TSL) approximation of the Navier-Stokes equations.

Let us consider for illustration the differential form of the Navier-Stokes equations formulated for a general curvilinear grid (A.4). If we further assume that, as rendered in Fig. 2.4, the wall is located at $\eta = \text{const.}$, all derivatives in the streamwise ($\partial/\partial\xi$) and in the cross-flow direction ($\partial/\partial\zeta$) are omitted from the diffusive terms. The 3-D TSL Navier-Stokes equations in differential form then appear as follows in a curvilinear coordinate system (ξ, η, ζ)

$$\frac{\partial \vec{W}^*}{\partial t} + \frac{\partial \vec{F}_{c,1}}{\partial \xi} + \frac{\partial \vec{F}_{c,2}}{\partial \eta} + \frac{\partial \vec{F}_{c,3}}{\partial \zeta} = \frac{\partial \vec{F}_{v,2}}{\partial \eta} + \vec{Q}^* \,. \tag{A.39}$$

The vectors of the conservative variables ($\vec{W}^*$), of the convective fluxes ($\vec{F}_{c,1/2/3}$), and of the diffusive flux ($\vec{F}_{v,2}$), respectively, are given by the relationships (A.5)-(A.7). Furthermore, the source term ($\vec{Q}^*$) also stays in the form given by Eq. (A.8). However, the components of the viscous stress tensor (Eq. (A.10)) and of the thermal fluxes transform to

$$\begin{aligned}
\tau_{xx} &= 2\mu\,\eta_x\frac{\partial u}{\partial \eta} + \lambda\left(\eta_x\frac{\partial u}{\partial \eta} + \eta_y\frac{\partial v}{\partial \eta} + \eta_z\frac{\partial w}{\partial \eta}\right)\\
\tau_{yy} &= 2\mu\,\eta_y\frac{\partial v}{\partial \eta} + \lambda\left(\eta_x\frac{\partial u}{\partial \eta} + \eta_y\frac{\partial v}{\partial \eta} + \eta_z\frac{\partial w}{\partial \eta}\right)\\
\tau_{zz} &= 2\mu\,\eta_z\frac{\partial w}{\partial \eta} + \lambda\left(\eta_x\frac{\partial u}{\partial \eta} + \eta_y\frac{\partial v}{\partial \eta} + \eta_z\frac{\partial w}{\partial \eta}\right)\\
\tau_{xy} &= \tau_{yx} = \mu\left(\eta_y\frac{\partial u}{\partial \eta} + \eta_x\frac{\partial v}{\partial \eta}\right)\\
\tau_{xz} &= \tau_{zx} = \mu\left(\eta_z\frac{\partial u}{\partial \eta} + \eta_x\frac{\partial w}{\partial \eta}\right)\\
\tau_{yz} &= \tau_{zy} = \mu\left(\eta_z\frac{\partial v}{\partial \eta} + \eta_y\frac{\partial w}{\partial \eta}\right)\\
\Theta_x &= u\tau_{xx} + v\tau_{xy} + w\tau_{xz} + k\,\eta_x\frac{\partial T}{\partial \eta}\\
\Theta_y &= u\tau_{yx} + v\tau_{yy} + w\tau_{yz} + k\,\eta_y\frac{\partial T}{\partial \eta}\\
\Theta_z &= u\tau_{zx} + v\tau_{zy} + w\tau_{zz} + k\,\eta_z\frac{\partial T}{\partial \eta}\,.
\end{aligned} \tag{A.40}$$

In the above formulae (A.40), the partial derivatives with respect to the Cartesian coordinates were approximated as

$$\begin{aligned} \frac{\partial u}{\partial x} &\approx \eta_x \frac{\partial u}{\partial \eta} \\ \frac{\partial u}{\partial x} &\approx \eta_y \frac{\partial u}{\partial \eta} \\ \frac{\partial u}{\partial y} &\approx \eta_z \frac{\partial u}{\partial \eta}\,, \quad \text{etc.} \end{aligned} \tag{A.41}$$

accordingly to the TSL assumption.

A.6 Parabolised Navier-Stokes Equations

In certain cases, where the following three conditions are met:

- the flow is steady (i.e., $\partial \vec{W}/\partial t{=}0$),
- the fluid moves predominantly in one main direction (there must be no boundary layer separation),
- the cross-flow components are negligible,

the Navier-Stokes equations (2.19)) can be simplified to a form called the *Parabolised Navier-Stokes* (PNS) equations [19]. Then, the derivatives of the velocity components and of the temperature with respect to the streamwise direction can be dropped from the viscous stress terms. Furthermore, the components of the viscous stress tensor $\overline{\overline{\tau}}$ and of its work $(\overline{\overline{\tau}} \cdot \vec{v})$ in the streamwise direction can be neglected in the viscous flux vector in Eq. (2.23).

In order to demonstrate the PNS approach, let us consider the differential form of the Navier-Stokes equations written for a general curvilinear grid as given by Eq. (A.4). Let us also assume that the streamwise direction corresponds to the coordinate ξ and the cross-flow directions to the coordinates η and ζ, respectively. Then, the differential form of the 3-D PNS equations reads in a curvilinear coordinate system (ξ, η, ζ) as

$$\frac{\partial \vec{W}^*}{\partial t} + \frac{\partial \vec{F}_{c,1}}{\partial \xi} + \frac{\partial \vec{F}_{c,2}}{\partial \eta} + \frac{\partial \vec{F}_{c,3}}{\partial \zeta} = \frac{\partial \vec{F}_{v,2}}{\partial \eta} + \frac{\partial \vec{F}_{v,3}}{\partial \zeta} + \vec{Q}^* . \tag{A.42}$$

Hence, the viscous flux in the streamwise direction $(\vec{F}_{v,1})$ was assumed to be zero. The partial derivatives of the Cartesian velocity components and of the temperature, which appear in the viscous stress tensor and in the thermal fluxes (A.10), are approximated as

$$\begin{aligned}
\frac{\partial u}{\partial x} &\approx \eta_x \frac{\partial u}{\partial \eta} + \zeta_x \frac{\partial u}{\partial \zeta} \\
\frac{\partial u}{\partial y} &\approx \eta_y \frac{\partial u}{\partial \eta} + \zeta_y \frac{\partial u}{\partial \zeta} \\
\frac{\partial u}{\partial z} &\approx \eta_z \frac{\partial u}{\partial \eta} + \zeta_z \frac{\partial u}{\partial \zeta} \,, \quad \text{etc.}
\end{aligned} \tag{A.43}$$

within the PNS approach instead of using the exact formulae (A.12). The definition of the conservative variables (A.5), the relations for the convective fluxes (A.6), for the two remaining viscous fluxes (A.7) as well as for the source term (A.8) stay all unchanged.

A.7 Convective Flux Jacobian

The convective flux Jacobian represents the gradient of the convective fluxes with respect to the conservative variables, i.e.,

$$\bar{A}_c = \frac{\partial \vec{F}_c}{\partial \vec{W}}\,, \tag{A.44}$$

where $\vec{F}_c$ is given by Eq. (2.21).

2-D Formulation

The Jacobian matrix of the convective fluxes takes in two dimensions the following form [20]

$$\bar{A}_c = \begin{bmatrix} -V_t & n_x & n_y & 0 \\ n_x\phi - uV & V - V_t - a_3 n_x u & n_y u - a_2 n_x v & a_2 n_x \\ n_y \phi - vV & n_x v - a_2 n_y u & V - V_t - a_3 n_y v & a_2 n_y \\ V(\phi - a_1) & n_x a_1 - a_2 uV & n_y a_1 - a_2 vV & \gamma V - V_t \end{bmatrix}, \tag{A.45}$$

where

$$\begin{aligned} a_1 &= \gamma E - \phi \\ a_2 &= \gamma - 1 \\ a_3 &= \gamma - 2 \\ V &= n_x u + n_y v \\ \phi &= \tfrac{1}{2}(\gamma - 1)(u^2 + v^2) \end{aligned} \tag{A.46}$$

and n_x, n_y denote the Cartesian components of the unit normal vector $\vec{n}$ (Fig. 2.1). Furthermore, γ stands for the ratio of specific heat coefficients and V represents the contravariant velocity. The contravariant velocity of the face of the control volume (V_t - see Eq. (A.35)) is set to zero for stationary grids.

3-D Formulation

The expression for the convective flux Jacobian reads in three dimensions [20]

$$\bar{A}_c = \begin{bmatrix} -V_t & n_x & n_y & n_z & 0 \\ n_x\phi - uV & V - V_t - a_3 n_x u & n_y u - a_2 n_x v & n_z u - a_2 n_x w & a_2 n_x \\ n_y\phi - vV & n_x v - a_2 n_y u & V - V_t - a_3 n_y v & n_z v - a_2 n_y w & a_2 n_y \\ n_z\phi - wV & n_x w - a_2 n_z u & n_y w - a_2 n_z v & V - V_t - a_3 n_z w & a_2 n_z \\ V(\phi - a_1) & n_x a_1 - a_2 uV & n_y a_1 - a_2 vV & n_z a_1 - a_2 wV & \gamma V - V_t \end{bmatrix} \tag{A.47}$$

with the abbreviations

$$\begin{aligned}
a_1 &= \gamma E - \phi \\
a_2 &= \gamma - 1 \\
a_3 &= \gamma - 2 \\
V &= n_x u + n_y v + n_z w \\
\phi &= \tfrac{1}{2}(\gamma - 1)\big(u^2 + v^2 + w^2\big)\,.
\end{aligned} \tag{A.48}$$

In the above relations (A.47) and (A.48), respectively, n_x, n_y, and n_z denote the Cartesian components of the unit normal vector $\vec{n}$ (see Fig. 2.1). In the case of stationary grids, we have to set $V_t = 0$.

A.8 Viscous Flux Jacobian

The viscous flux Jacobian represents the gradient of the viscous fluxes with respect to the conservative variables, i.e.,

$$\bar{A}_v = \frac{\partial \vec{F}_v}{\partial \vec{W}}, \tag{A.49}$$

where $\vec{F}_v$ is given by Eq. (2.23). In the following, we shall consider for simplicity the Thin Shear Layer (TSL) approximation of the Navier-Stokes equations only (cf. Subsection 2.4.3 and Section A.5). Furthermore, it is assumed that the dynamic viscosity and the thermal conductivity coefficients are frozen with respect to changes in the conservative variables and in space.

2-D Formulation

The 2-D viscous flux Jacobian reads in curvilinear coordinates for the TSL approximation as [20]

$$\bar{A}_v = \frac{\mu}{J}\begin{bmatrix} 0 & 0 & 0 & 0 \\ b_{21} & a_1\partial_\psi(\rho^{-1}) & a_2\partial_\psi(\rho^{-1}) & 0 \\ b_{31} & a_2\partial_\psi(\rho^{-1}) & a_3\partial_\psi(\rho^{-1}) & 0 \\ b_{41} & b_{42} & b_{43} & a_4\partial_\psi(\rho^{-1}) \end{bmatrix}, \tag{A.50}$$

where J represents the determinant of the coordinate transformation Jacobian correspondingly to Eq. (A.13). The coefficients a and b are given by

$$\begin{aligned} b_{21} &= -a_1\partial_\psi(u/\rho) - a_2\partial_\psi(v/\rho) \\ b_{31} &= -a_2\partial_\psi(u/\rho) - a_3\partial_\psi(v/\rho) \\ b_{41} &= a_4\partial_\psi\left[(u^2+v^2)/\rho - E/\rho\right] - a_1\partial_\psi(u^2/\rho) - \\ &\quad\ 2a_2\partial_\psi(uv/\rho) - a_3\partial_\psi(v^2/\rho) \\ b_{42} &= -a_4\partial_\psi(u/\rho) - b_{21} \\ b_{43} &= -a_4\partial_\psi(v/\rho) - b_{31} \\ a_1 &= (4/3)\psi_x^2 + \psi_y^2 \\ a_2 &= (1/3)\psi_x\psi_y \\ a_3 &= \psi_x^2 + (4/3)\psi_y^2 \\ a_4 &= (\gamma/Pr)\left(\psi_x^2 + \psi_y^2\right). \end{aligned} \tag{A.51}$$

In the above relations, μ stands for the dynamic viscosity coefficient, γ for the ratio of specific heat coefficients and Pr denotes the Prandtl number, respectively.

The terms $\partial_\psi() = \partial()/\partial\psi$ must be conceived as operators. Let us consider for illustration the term $a_1\partial_\psi(\rho^{-1})$ in Eq. (A.50). Within many implicit schemes, the viscous flux Jacobian $\bar{A}_v$ is multiplied by the change of the conservative variables $\Delta\vec{W} = \vec{W}^{n+1} - \vec{W}^n$. Hence, the complete term would read

$$\left[J^{-1}\mu\, a_1\partial_\psi(\rho^{-1})\right]\Delta W_2 = A_{22}\,\Delta W_2\,. \tag{A.52}$$

If we discretise now the above term on structured grid using backward differences, we obtain at the mid-point $(i+1/2)$

$$\left[A_{22}\,\Delta W_2\right]_{i+1/2} = J^{-1}_{i+1/2}\,\mu_{i+1/2}\,(a_1)_{i+1/2}\,\frac{\left(\dfrac{\Delta W_2}{\rho}\right)_{i+1} - \left(\dfrac{\Delta W_2}{\rho}\right)_i}{\Delta\psi}\,. \tag{A.53}$$

Jacobian matrices for specific coordinate directions result if we set $\psi = \xi$ or $\psi = \eta$, respectively.

3-D Formulation

The 3-D viscous flux Jacobian takes in curvilinear coordinates the following form for the TSL approximation [20]

$$\bar{A}_v = \frac{\mu}{J}\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ b_{21} & a_1\partial_\psi(\rho^{-1}) & a_2\partial_\psi(\rho^{-1}) & a_3\partial_\psi(\rho^{-1}) & 0 \\ b_{31} & a_2\partial_\psi(\rho^{-1}) & a_4\partial_\psi(\rho^{-1}) & a_5\partial_\psi(\rho^{-1}) & 0 \\ b_{41} & a_3\partial_\psi(\rho^{-1}) & a_5\partial_\psi(\rho^{-1}) & a_6\partial_\psi(\rho^{-1}) & 0 \\ b_{51} & b_{52} & b_{53} & b_{54} & a_7\partial_\psi(\rho^{-1}) \end{bmatrix}, \tag{A.54}$$

where

$$\begin{aligned} b_{21} &= -a_1\partial_\psi(u/\rho) - a_2\partial_\psi(v/\rho) - a_3\partial_\psi(w/\rho) \\ b_{31} &= -a_2\partial_\psi(u/\rho) - a_4\partial_\psi(v/\rho) - a_5\partial_\psi(w/\rho) \\ b_{41} &= -a_3\partial_\psi(u/\rho) - a_5\partial_\psi(v/\rho) - a_6\partial_\psi(w/\rho) \\ b_{51} &= a_7\partial_\psi\left[(u^2+v^2+w^2)/\rho - E/\rho\right] - \\ &\quad a_1\partial_\psi(u^2/\rho) - a_4\partial_\psi(v^2/\rho) - a_6\partial_\psi(w^2/\rho) - \\ &\quad 2a_2\partial_\psi(uv/\rho) - 2a_3\partial_\psi(uw/\rho) - 2a_5\partial_\psi(vw/\rho) \\ b_{52} &= -a_7\partial_\psi(u/\rho) - b_{21} \\ b_{53} &= -a_7\partial_\psi(v/\rho) - b_{31} \\ b_{54} &= -a_7\partial_\psi(w/\rho) - b_{41} \end{aligned} \tag{A.55}$$

and

$$\begin{aligned}
a_1 &= (4/3)\psi_x^2 + \psi_y^2 + \psi_z^2 \\
a_2 &= (1/3)\psi_x\psi_y \\
a_3 &= (1/3)\psi_x\psi_z \\
a_5 &= (1/3)\psi_y\psi_z \\
a_4 &= \psi_x^2 + (4/3)\psi_y^2 + \psi_z^2 \\
a_6 &= \psi_x^2 + \psi_y^2 + (4/3)\psi_z^2 \\
a_7 &= (\gamma/Pr)\left(\psi_x^2 + \psi_y^2 + \psi_z^2\right) .
\end{aligned} \tag{A.56}$$

Again, the terms $\partial_\psi() = \partial()/\partial\psi$ have to be treated as operators. Jacobian matrices for specific coordinate directions are obtained by setting $\psi = \xi$ or $\psi = \eta$ or $\psi = \zeta$, respectively.

In the case of a finite volume discretisation, the metric terms ψ_x, ψ_y and ψ_z, can be transformed into the components of the face vector $\vec{S} = \vec{n}\Delta S$ according to Eq. (A.17). Furthermore, the determinant of the Jacobian of the coordinate transformation J is replaced by the inverse of the volume formed around the face of the control volume as indicated by Fig. A.1 and by Eq. (A.20). Thus, if we repeat the example of Eq. (A.52) for a finite volume scheme, the product $J^{-1}a_1$ in Eq. (A.51) will read

$$J^{-1}a_1 = \frac{2}{\Omega^*}\left(\frac{4}{3}S_x^2 + S_y^2\right) . \tag{A.57}$$

It should be noted that the face area was included in Eq. (A.57) to reflect the multiplication of the viscous fluxes with ΔS when the residual is computed – $\vec{F}_v$ itself contains only the components of the unit normal vector. The discretisation at the mid-point $(i+1/2)$ then becomes

$$\left[A_{22}\,\Delta W_2\right]_{i+1/2} = \frac{2\,\mu_{i+1/2}}{\Omega^*_{i+1/2}}\left(\frac{4}{3}S_x^2 + S_y^2\right)\left[\left(\frac{\Delta W_2}{\rho}\right)_{i+1} - \left(\frac{\Delta W_2}{\rho}\right)_i\right] , \tag{A.58}$$

where we assumed $\Delta\psi = 1$ due to the definition of $\vec{S}^I$ and $\vec{S}^J$ (see Fig. A.1 and Equations (A.15), (A.16)).

A.9 Transformation from Conservative to Characteristic Variables

The convective flux Jacobian (Section A.7) has real eigenvalues and a complete set of eigenvectors. Therefore, the Jacobians can be diagonalised [21] as

$$\bar{A}_c = \bar{T}\bar{\Lambda}_c\bar{T}^{-1}\,, \tag{A.59}$$

where $\bar{T}^{-1}$ denotes the matrix of left eigenvectors, $\bar{T}$ of right eigenvectors and $\bar{\Lambda}_c$ represents the diagonal matrix of eigenvalues, respectively. The transformation from conservative ($\vec{W}$) to characteristic variables ($\vec{C}$) reads

$$\vec{C} = \bar{T}^{-1}\vec{W}\,. \tag{A.60}$$

The diagonalisation of the convective flux Jacobian can be viewed as a decomposition into different waves. The right eigenvectors represent the waves, the characteristic variables are the wave amplitudes, and finally the eigenvalues are the associated wave speeds. In the context of the diagonalisation that will be presented below, we differentiate between two types of waves. First, there are *convective* or *linear* waves, which are connected to eigenvalues of the type $\Lambda_c = \vec{v}\cdot\vec{n}$. Second, there are *acoustic* waves which are related to eigenvalues of the type $\Lambda_c = \vec{v}\cdot\vec{n} \pm c$. It should be noted that the diagonalisation cannot be conducted simultaneously in multiple spatial directions [22]. This is the reason why the flux-difference splitting and the TVD schemes employ only 1-D wave decomposition. Only the fluctuation-splitting schemes (see Subsection 3.1.5) use an **approximate** multidimensional decomposition.

2-D Formulation

The matrix of the left eigenvectors of $\bar{A}_c$ appears as follows in two dimensions [20]

$$\bar{T}^{-1} = \begin{bmatrix} (1-\phi c^{-2}) & a_1 u\, c^{-2} & a_1 v\, c^{-2} & -a_1 c^{-2} \\ -(n_x u - n_y v)\rho^{-1} & n_y \rho^{-1} & -n_x \rho^{-1} & 0 \\ a_2(\phi - cV) & a_2(n_x c - a_1 u) & a_2(n_y c - a_1 v) & a_1 a_2 \\ a_2(\phi + cV) & -a_2(n_x c + a_1 u) & -a_2(n_y c + a_1 v) & a_1 a_2 \end{bmatrix}. \tag{A.61}$$

The matrix of the right eigenvectors takes the form [20]

$$\bar{T} = \begin{bmatrix} 1 & 0 & a_3 & a_3 \\ u & n_y\rho & a_3(u + n_x c) & a_3(u - n_x c) \\ v & -n_x\rho & a_3(v + n_y c) & a_3(v - n_y c) \\ \phi\, a_1^{-1} & \rho(n_y u - n_x v) & a_3(a_4 + cV) & a_3(a_4 - cV) \end{bmatrix}. \tag{A.62}$$

The diagonal matrix contains the real eigenvalues

$$\bar{\Lambda}_c = \begin{bmatrix} \Lambda_1 & 0 & 0 & 0 \\ 0 & \Lambda_2 & 0 & 0 \\ 0 & 0 & \Lambda_3 & 0 \\ 0 & 0 & 0 & \Lambda_4 \end{bmatrix} . \tag{A.63}$$

The following definitions apply in the above relationships

$$\begin{aligned} a_1 &= \gamma - 1 \\ a_2 &= \frac{1}{\rho c\sqrt{2}} \\ a_3 &= \frac{\rho}{c\sqrt{2}} \\ a_4 &= \frac{\phi + c^2}{\gamma - 1} \\ V &= n_x u + n_y v \\ \phi &= \tfrac{1}{2}(\gamma - 1)(u^2 + v^2) \\ \Lambda_1 &= \Lambda_2 = V - V_t \\ \Lambda_3 &= V - V_t + c \\ \Lambda_4 &= V - V_t - c . \end{aligned} \tag{A.64}$$

Furthermore, γ denotes the ratio of specific heat coefficients, c the speed of sound, $\vec{n} = [n_x, n_y]^T$ the unit normal vector, and V the contravariant velocity, respectively. Finally, V_t represents the contravariant velocity of the face of the control volume as given by Eq. (A.35). In the case of stationary grids, V_t has to be set to zero.

3-D Formulation

The matrix of the left eigenvectors of $\bar{A}_c$ becomes in three dimensions [20]

$$\bar{T}^{-1} = \begin{bmatrix} n_x a_5 - (n_z v - n_y w)\rho^{-1} & n_x a_1 u\, c^{-2} & n_x a_1 v\, c^{-2} + n_z \rho^{-1} \\ n_y a_5 - (n_x w - n_z u)\rho^{-1} & n_y a_1 u\, c^{-2} - n_z \rho^{-1} & n_y a_1 v\, c^{-2} \\ n_z a_5 - (n_y u - n_x v)\rho^{-1} & n_z a_1 u\, c^{-2} + n_y \rho^{-1} & n_z a_1 v\, c^{-2} - n_x \rho^{-1} \\ a_2(\phi - cV) & -a_2(a_1 u - n_x c) & -a_2(a_1 v - n_y c) \\ a_2(\phi + cV) & -a_2(a_1 u + n_x c) & -a_2(a_1 v + n_y c) \end{bmatrix}$$
$$\begin{matrix} n_x a_1 w\, c^{-2} - n_y \rho^{-1} & -n_x a_1 c^{-2} \\ n_y a_1 w\, c^{-2} - n_x \rho^{-1} & -n_y a_1 c^{-2} \\ n_z a_1 w\, c^{-2} & -n_z a_1 c^{-2} \\ -a_2(a_1 w - n_z c) & a_1 a_2 \\ -a_2(a_1 w + n_z c) & a_1 a_2 \end{matrix} \Bigg] . \tag{A.65}$$

The matrix of the right eigenvectors reads as [20]

$$\bar{T} = \begin{bmatrix} n_x & n_y & n_z & a_3 & a_3 \\ n_x u & n_y u - n_z \rho & n_z u + n_y \rho & a_3(u + n_x c) & a_3(u - n_x c) \\ n_x v + n_z \rho & n_y v & n_z v - n_x \rho & a_3(v + n_y c) & a_3(v - n_y c) \\ n_x w - n_y \rho & n_y w + n_x \rho & n_z w & a_3(w + n_z c) & a_3(w - n_z c) \\ n_x a_6 + \rho(n_z v - n_y w) & n_y a_6 + \rho(n_x w - n_z u) & n_z a_6 + \rho(n_y u - n_x v) & a_3(a_4 + cV) & a_3(a_4 - cV) \end{bmatrix} . \tag{A.66}$$

The diagonal matrix contains the real eigenvalues

$$\bar{\Lambda}_c = \begin{bmatrix} \Lambda_1 & 0 & 0 & 0 & 0 \\ 0 & \Lambda_2 & 0 & 0 & 0 \\ 0 & 0 & \Lambda_3 & 0 & 0 \\ 0 & 0 & 0 & \Lambda_4 & 0 \\ 0 & 0 & 0 & 0 & \Lambda_5 \end{bmatrix} . \tag{A.67}$$

The following abbreviations were used

$$\begin{aligned} a_1 &= \gamma - 1 \\ a_2 &= \frac{1}{\rho c \sqrt{2}} \\ a_3 &= \frac{\rho}{c \sqrt{2}} \\ a_4 &= \frac{\phi + c^2}{\gamma - 1} \\ a_5 &= 1 - \frac{\phi}{c^2} \\ a_6 &= \frac{\phi}{\gamma - 1} \\ V &= n_x u + n_y v + n_z w \\ \phi &= \tfrac{1}{2}(\gamma - 1)\left(u^2 + v^2 + w^2\right) \\ \Lambda_1 &= \Lambda_2 = \Lambda_3 = V - V_t \\ \Lambda_4 &= V - V_t + c \\ \Lambda_5 &= V - V_t - c\,. \end{aligned} \tag{A.68}$$

In the above relations Eqs. (A.65)-(A.68), γ denotes the ratio of specific heat coefficients, c the speed of sound, $\vec{n} = [n_x, n_y, n_z]^T$ the unit normal vector, and V the contravariant velocity, respectively. Finally, V_t represents the contravariant velocity of the face of the control volume as given by Eq. (A.35). In the case of stationary grids, V_t has to be set to zero.

A.10 GMRES Algorithm

Consider the system of linear equations

$$\bar{A}\vec{x} = \vec{b}. \tag{A.69}$$

We are looking for an approximate solution of the form

$$\vec{x} = \vec{x}_0 + \vec{z}, \tag{A.70}$$

where $\vec{x}_0$ represents an initial guess and $\vec{z}$ is a member of the Krylov subspace

$$\vec{z} \in \mathcal{K}_m, \quad \mathcal{K}_m \equiv \text{span}\left\{\vec{r}_0,\, \bar{A}\vec{r}_0,\, \bar{A}^2\vec{r}_0,\, \ldots,\, \bar{A}^{m-1}\vec{r}_0\right\} \tag{A.71}$$

with $\vec{r}_0 = \vec{b} - \bar{A}\vec{x}_0$, and m being the dimension of $\mathcal{K}$. The parameter m is also termed the number of search directions. The *Generalised Minimal Residual* (GMRES) algorithm [23] determines $\vec{z}$ in such a way that the 2-norm of the residual, i.e.,

$$\| \vec{b} - \bar{A}(\vec{x}_0 + \vec{z}) \| \tag{A.72}$$

is minimised. In the following, we present the particular steps of the GMRES algorithm.

1. Computation of the orthonormal basis of $\mathcal{K}_m$

We employ the modified Gram-Schmidt procedure

$$\vec{r}_0 = \vec{b} - \bar{A}\vec{x}_0$$
$$\vec{v}_1 = \vec{r}_0 / \| \vec{r}_0 \|$$

DO $j = 1, m$

$\quad \vec{\tilde{v}}_{j+1} = \bar{A}\vec{v}_j$

$\quad$ DO $i = 1, j$

$\quad\quad h_{i,j} = \vec{\tilde{v}}_{j+1} \cdot \vec{v}_i$

$\quad\quad \vec{\tilde{v}}_{j+1} = \vec{\tilde{v}}_{j+1} - h_{i,j}\,\vec{v}_i$

$\quad$ ENDDO

$\quad h_{j+1,j} = \| \vec{\tilde{v}}_{j+1} \|$

$\quad \vec{v}_{j+1} = \vec{\tilde{v}}_{j+1} / h_{j+1,j}$

ENDDO

where $h_{i,j}$ denotes the coefficients of the upper Hessenberg matrix (i = line, j = column). However, the matrix is extended by the elements $h_{j+1,j}$. Therefore, the dimensions becomes $(m+1) \times m$.

2. Generation of the upper Hessenberg matrix

The matrix has the following almost triangular form

$$\bar{H}_m^* = \begin{bmatrix} h_{1,1} & h_{1,2} & \cdots & h_{1,m-1} & h_{1,m} \\ h_{2,1} & h_{2,2} & \cdots & h_{2,m-1} & h_{2,m} \\ 0 & h_{3,2} & \ddots & \vdots & \vdots \\ \vdots & 0 & \ddots & h_{m-1,m-1} & h_{m-1,m} \\ \vdots & \vdots & \ddots & h_{m,m-1} & h_{m,m} \\ 0 & 0 & \cdots & 0 & h_{m+1,m} \end{bmatrix}^{(m+1)\times m} \tag{A.73}$$

It is used further below to formulate and solve the minimisation problem for the residual (Eq. (A.72)).

3. Minimisation of the residual

The correction of the start solution $\vec{x}_0$ is defined as [23]

$$\vec{z} = \sum_{j=1}^{m} y_j \vec{v}_j \,, \tag{A.74}$$

where y_j are the components of the vector

$$\vec{y} = [\,y_1,\, y_2,\, \cdots,\, y_m\,]^T \,. \tag{A.75}$$

Furthermore, it can be shown that

$$\bar{A}\bar{V}_m = \bar{V}_{m+1}\bar{H}_m^* \tag{A.76}$$

with

$$\bar{V}_m = [\,\vec{v}_1,\, \vec{v}_2,\, \cdots,\, \vec{v}_m\,] \tag{A.77}$$

being a matrix with $\vec{v}_j$ as columns. Let us introduce the notation

$$\vec{e} = [\,\|\,\vec{r}_0\,\|,\, 0,\, \cdots,\, 0]^T \,, \tag{A.78}$$

where $\vec{e}$ has $(m+1)$ elements. Using the definition of Eq. (A.78), we observe that $\vec{r}_0 = \vec{b} - \bar{A}\vec{x}_0 = \bar{V}_{m+1}\vec{e}$. Hence, we obtain for the residual Eq. (A.72)

$$\begin{aligned} \|\,\vec{b} - \bar{A}(\vec{x}_0 + \vec{z})\,\| &= \|\,\vec{r}_0 - \bar{A}\big(\textstyle\sum_{j=1}^{m} y_j\vec{v}_j\big)\,\| \\ &= \|\,\vec{r}_0 - \bar{A}\bar{V}_m\,\vec{y}\,\| \\ &= \|\,\bar{V}_{m+1}(\vec{e} - \bar{H}_m^*\,\vec{y})\,\| \\ &= \|\,\vec{e} - \bar{H}_m^*\,\vec{y}\,\| \,. \end{aligned} \tag{A.79}$$

We employed the orthonormality of $\bar{V}_{m+1}$ in the last step (this means $\vec{v}_i^T \cdot \vec{v}_j = 0$ for $i \neq j$ and $\vec{v}_i^T \cdot \vec{v}_j = 1$ for $i = j$). Therefore, the problem of the minimisation of the residual can be simplified as

$$\min_{\vec{z} \in \mathcal{K}_m} \| \vec{b} - \bar{A}(\vec{x}_0 + \vec{z}) \| = \min_{\vec{y} \in \mathcal{R}^m} \| \vec{e} - \bar{H}_m^* \vec{y} \| \; . \tag{A.80}$$

The solution of the minimisation problem can be obtained with the help of the Q-R algorithm which is described next.

4. Q-R algorithm

Let us define $\bar{R}_m = \bar{Q}_m \bar{H}_m^*$ with

$$\bar{Q}_m \stackrel{\text{def}}{=} \bar{F}_m \bar{F}_{m-1} \cdots \bar{F}_1 \tag{A.81}$$

being the product of the Givens rotation matrices

$$\bar{F}_j = \begin{bmatrix} \bar{I}_{j-1} & & \\ & \begin{bmatrix} c_j & s_j \\ -s_j & c_j \end{bmatrix} & \\ & & \bar{I}_{m-j} \end{bmatrix}^{(m+1)\times(m+1)} \; . \tag{A.82}$$

In Eq. (A.82), $\bar{I}_j$ denotes the identity matrix of dimension j. Further, c_j and s_j ($c_j^2 + s_j^2 = 1$) represent the sine/cosine of the rotation angle. The rotations are chosen such that $\bar{H}_m^*$ is transformed into an upper triangular matrix $\bar{R}_m$ which has the dimensions $(m+1) \times m$ and which last line contains only zeros. Since $\bar{Q}_m^T \bar{Q}_m = \bar{I}$, we can write in Eq. (A.80)

$$\begin{aligned} \| \vec{e} - \bar{H}_m^* \vec{y} \| &= \| \bar{Q}_m^T (\bar{Q}_m \vec{e} - \bar{Q}_m \bar{H}_m^* \vec{y}) \| \\ &= \| \vec{g} - \bar{R}_m \vec{y} \| \, , \end{aligned} \tag{A.83}$$

where $\vec{g} = \bar{Q}_m \vec{e}$ denotes the transformation of the vector $\vec{e}$ (Eq. (A.78)). The last line of $\bar{R}_m$ consists of zeros, therefore only the term g_{m+1} is nonzero in the row $(m+1)$ of the vector $(\vec{g} - \bar{R}_m \vec{y})$. If we denote the first m-components of $(\vec{g} - \bar{R}_m \vec{y})$ as p_j ($j = 1, \cdots, m$), then the norm in Eq. (A.83) becomes

$$\| \vec{g} - \bar{R}_m \vec{y} \| = \sqrt{g_{m+1}^2 + \sum_{j=1}^{m} p_j^2} \, . \tag{A.84}$$

If we chose the components y_j of $\vec{y}$ in such a way that $p_j^2 = 0$ for all $j = 1, \cdots, m$, we obtain for the minimisation problem in Eq. (A.80)

$$\min_{\vec{y} \in \mathcal{R}^m} \| \vec{g} - \bar{R}_m \vec{y} \| = |g_{m+1}| \, . \tag{A.85}$$

The components y_j results from the solution of the following system of linear equations

$$\begin{bmatrix} R_{1,1} & \cdots & R_{1,m-1} & R_{1,m} \\ 0 & \ddots & \vdots & \vdots \\ \vdots & \ddots & R_{m-1,m-1} & R_{m-1,m} \\ 0 & \cdots & 0 & R_{m,m} \end{bmatrix} \begin{bmatrix} y_1 \\ \vdots \\ y_{m-1} \\ y_m \end{bmatrix} = \begin{bmatrix} g_1 \\ \vdots \\ g_{m-1} \\ g_m \end{bmatrix} . \qquad \text{(A.86)}$$

by back-substitution. The solution of the system of equations (A.69) is then obtained with known y_j from Eq. (A.74).

It is important to remark that

$$\begin{aligned} \| \vec{r}_m \| &= \| \vec{b} - \bar{A}(\vec{x}_0 + \vec{z}) \| \\ &= \| \vec{e} - \bar{H}^*_m \vec{y} \| \\ &= \| \vec{g} - \bar{R}_m \vec{y} \| \\ &= |g_{m+1}| \, . \end{aligned} \qquad \text{(A.87)}$$

This means that the actual residual can be easily determined as $|g_{m+1}|$.

A.11 Tensor Notation

Expressions like coordinate (x_i) or velocity components (v_i) represent *first-order tensors*. They have three components and thus correspond to vectors. Hence,

$$\begin{aligned} x_i &= [x_1, x_2, x_3] = [x, y, z] = \vec{r} \\ v_i &= [v_1, v_2, v_3] = [u, v, w] = \vec{v}\,. \end{aligned} \tag{A.88}$$

Second-order tensors consist of nine components and can be written as 3×3 matrices, e.g.,

$$v_i v_j \equiv \begin{bmatrix} v_1 v_1 & v_1 v_2 & v_1 v_3 \\ v_2 v_1 & v_2 v_2 & v_2 v_3 \\ v_3 v_1 & v_3 v_2 & v_3 v_3 \end{bmatrix}. \tag{A.89}$$

Similarly, the tensor of viscous stresses $\overline{\overline{\tau}} = \tau_{ij}$ reads

$$\tau_{ij} \equiv \begin{bmatrix} \tau_{11} & \tau_{12} & \tau_{13} \\ \tau_{21} & \tau_{22} & \tau_{23} \\ \tau_{31} & \tau_{32} & \tau_{33} \end{bmatrix}. \tag{A.90}$$

The Kronecker symbol δ_{ij} is a special second-order tensor. It corresponds to a 3×3 identity matrix. Thus, the relation holds

$$\delta_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j\,. \end{cases} \tag{A.91}$$

The last important rule is the so-called *Einstein summation convention*. It states that whenever two identical indices occur in an expression, it means a sum over all three coordinate directions. With this, the scalar product between the vectors $\vec{u}$ and $\vec{v}$ can be expressed as

$$u_i v_i = u_1 v_1 + u_2 v_2 + u_3 v_3 = \vec{u} \cdot \vec{v}\,. \tag{A.92}$$

Furthermore, the divergence of the vector $\vec{v}$ becomes in tensor notation

$$\frac{\partial v_i}{\partial x_i} = \frac{\partial v_1}{\partial x_1} + \frac{\partial v_2}{\partial x_2} + \frac{\partial v_3}{\partial x_3} = \vec{\nabla} \cdot \vec{v}\,. \tag{A.93}$$

Bibliography

[1] Sommerfeld, A.J.W.: *Partial Differential Equations in Physics.* Academic, New York, 1949.

[2] Courant, R.; Hilbert,D.: *Methods of Mathematical Physics.* Interscience, New York, 1962.

[3] Hirsch, C.: *Numerical Computation of Internal and External Flows.* Vol. 1, John Wiley and Sons, 1988.

[4] Hirt, C.W.; Amsden, A.A.; Cook, J.L.: *An Arbitrary Lagrangian-Eulerian Computing Method for All Flow Speeds.* J. Computational Physics, 14 (1974), pp. 227-253.

[5] Pracht, W.E.: *Calculating Three-Dimensional Fluid Flows at All Speeds with an Eulerian-Lagrangian Computing Mesh.* J. Computational Physics, 17 (1975), pp. 132-159.

[6] Donea, J.: *An Arbitrary Lagrangian-Eulerian Finite Element Method for Transient Fluid-Structure Interactions.* Comp. Meths. Appl. Mech. Engrg., 33 (1982), pp. 689-723.

[7] Batina, J.T.: *Unsteady Euler Airfoil Solutions Using Unstructured Dynamic Meshes.* AIAA Paper 89-0115, 1989.

[8] Thomas, P.D.; Lombard, C.K.: *Geometric Conservation Law and Its Application to Flow Computations on Moving Grids.* AIAA Journal, 17 (1979), pp. 1030-1037.

[9] Tamura, Y.; Fujii, K.: *Conservation Law for Moving and Transformed Grids.* AIAA Paper 93-3365, 1993.

[10] Lesoinne, M.; Farhat, C.: *Geometric Conservation Laws for Flow Problems with Moving Boundaries and Deformable Meshes and their Impact on Aeroelastic Computations.* AIAA Paper 95-1709, 1995; also in Comp. Meth. Appl. Mech. Eng., 134 (1996), pp. 71-90.

[11] Guillard, H.; Farhat, C.: *On the Significance of the GCL for Flow Computations on Moving Meshes.* AIAA Paper 99-0793, 1999.

[12] Nkonga, B.; Guillard, H.: *Godunov Type Method on Non-Structured Meshes for Three-Dimensional Moving Boundary Problems.* Comp. Meths. Appl. Mech. Engrg., 113 (1994), pp. 183-204.

[13] Singh, K.P.; Newman, J.C.; Baysal, O.: *Dynamic Unstructured Method for Flows Past Multiple Objects in Relative Motion.* AIAA Journal, 33 (1995), pp. 641-649.

[14] Koobus, B.; Farhat, C.: *Time-Accurate Schemes for Computing Two- and Three-Dimensional Viscous Fluxes on Unstructured Dynamic Meshes.* INRIA Report No. 2833, March, 1996.

[15] Koobus, B.; Farhat, C.: *Second-Order Schemes that Satisfy the GCL for Flow Computations on Dynamic Grids.* AIAA Paper 98-0113, 1998.

[16] Venkatakrishnan, V.; Mavriplis, D.J.: *Implicit Method for the Computation of Unsteady Flows on Unstructured Grids.* J. Computational Physics, 127 (1996), pp. 380-397.

[17] Steger, J.L.: *Implicit Finite Difference Simulation of Flows About Arbitrary Geometries.* AIAA Journal 17 (1978) 679-686.

[18] Pulliam, T.H.; Steger, J.L.: *Implicit Finite Difference Simulations of 3-D Compressible Flows.* AIAA Journal 18 (1980) 159-167.

[19] Patankar, S.V.; Spalding, D.B.: *A Calculation Procedure for Heat, Mass and Momentum Transfer in Three-Dimensional Parabolic Flows.* Int. J. Heat Mass Transfer, 15 (1972), pp. 1787-1806.

[20] Pulliam, T.H.; Steger, J.L.: *Recent Improvements in Efficiency, Accuracy, and Convergence for Implicit Approximate Factorization Algorithms.* AIAA Paper 85-0360, 1985.

[21] Warming, R.F.; Beam, R.; Hyett, B.J.: *Diagonalization and Simultaneous Symmetrization of the Gas-Dynamic Matrices.* Math. Comp., 29 (1975), p. 1037.

[22] Whitham, G.B.: *Linear and Nonlinear Waves.* Wiley, New York, 1974.

[23] Saad, Y.; Schulz, M.H.: *GMRES: A Generalized Minimum Residual Algorithm for Solving Nonsymmetric Linear Systems.* SIAM J. Scientific and Statistical Computing, 7 (1986), pp. 856-869.

Index

ADI: Alternating Direction Implicit, 50, 199, 366
Advancing-front Delaunay, 367
Advancing-front method, 367, 373
Advancing-layers method, 376
Advancing-normal point placement, 379
Agglomeration multigrid, 316
Algebraic grid generation, 356, 359
Algebraic models, 55
All-speed flow, 320
AMG: Algebraic Multigrid, 305, 315
Amplification factor, 335
Approximate Riemann solver, 43, 105
Arrhenius formulae, 22
ARS: Algebraic Reynolds Stress, 54
Artificial compressibility method, 30, 320
Artificial dissipation, 41, 95, 150
AUSM: Advection Upstream Splitting Method, 42, 98, 101
Average of fluxes, 83, 88, 139, 142
Average of variables, 83, 88, 139, 145

Background grid, 368
Backscatter, 252
Baldwin-Lomax model, 55
Bi-CGSTAB: Bi-Conjugate Gradient Stabilised, 51, 208
Body-fitted grid, 29
Boundary conditions, 56
Boussinesq hypothesis, 54, 233
Bradshaw's assumption, 246
Bubble packing method, 367
Bulk viscosity, 15

Carbuncle phenomenon, 107
Cartesian grids, 29, 381
Cell-centred scheme, 37, 76, 83, 132, 139
Cell-vertex scheme, 37, 78, 85, 88, 132, 142
Central scheme, 41, 95, 150
Centrifugal force, 411
CFL condition, 186, 338, 347, 348
CFL number, 183, 186, 338, 347
CGS: Conjugate Gradient Squared, 51, 208
Characteristic boundary conditions, 56
Characteristic time-stepping, 47
Characteristic variables, 42, 106, 109, 115, 277, 424
Characteristics of PDE's, 407
Chimera technique, 34, 290
Circulation, 280
CIRS: Central Implicit Residual Smoothing, 301
Co-located grid scheme, 78
Coarse grid correction, 307
Computational molecule, 93
Computational space, 32, 356
Condition number, 320
Conjugate gradient method, 51, 208
Conservation laws, 5
Conservative variables, 16, 78
Consistency, 331, 332
Constrained Delaunay triangulation, 372
Containment-dual control volume, 146
Continuum, 5
Contravariant velocity, 16, 414
Control functions, 363

Control volume, 79, 80, 134, 135
Convective flux Jacobian, 419
Convective flux tensor, 7
Convective fluxes, 6, 16, 93, 150
Coordinate cut, 286, 357, 359
Coriolis force, 411
Curvilinear grid, 32
CUSP: Convective Upwind Split Pressure, 42, 98, 103
Cyclic directions, 292

Dalton's law, 21
Damköhler number, 19
Damping functions, 241, 243
Damping properties, 334
Decoupling, 95, 118, 171
Degrees of freedom, 39
Delaunay method, 367, 368
Density (mass) weighted decomposition, 53, 230
Density-based schemes, 30
Dependent variables, 78
Determinant of Jacobian of coordinate transformation, 404
Differential form, 401
Diffusive flux, 7
Diffusive flux tensor, 7
Dilatation, 15
Direct methods, 49, 191
Dirichlet tessellation, 368
Discretisation accuracy, 333
Discretisation error, 331
DNS: Direct Numerical Simulation, 53, 225
Domain decomposition, 34
Dual control volumes, 37, 78, 88, 142
Dual time-stepping approach, 49, 212
Dummy cells, 268
Dummy points, 268
Dynamic SGS models, 254
Dynamic viscosity coefficient, 13

Eddy viscosity, 55, 225, 233
Edge collapsing, 316
Eigenvalue, 106, 108, 425, 426
Eigenvector, 106, 109, 424–426
Einstein summation convention, 431
Elliptic grid generation, 356, 363
ENO: Essentially Non-Oscillatory, 43
Ensemble averaging, 230
Enthalpy, 18
Enthalpy damping, 300
Entropy condition, 38
Entropy correction, 108, 109
Equation of state, 18
Ergodic hypothesis, 230
Euler equations, 25
Explicit operator, 190
Explicit scheme, 182
Explicit time-stepping scheme, 46
External body forces, 9
External volume forces, 9

Face vector, 79–81, 135–137
Factored scheme, 191
Factorisation error, 50, 201
FAS: Full Approximation Storage, 48, 306
Favre (mass) averaging, 53, 230
Favre decomposition, 53, 230
Finite control volume, 6
Finite difference method, 32, 36
Finite element method, 32, 39
Finite volume method, 32, 37
First-order closures, 54, 225, 238
Fluctuation-splitting scheme, 42, 43
Fluid dynamics, 5
Flux, 5
Flux Jacobian, 32, 42, 47, 49, 190, 191, 195
Flux-difference splitting scheme, 42, 43, 105
Flux-vector splitting scheme, 42, 98
FMG: Full Multigrid, 48, 309
Forcing function, 306
Fourier symbol, 335
Frontal Delaunay, 367

Gauss's theorem, 401
Gaussian filter, 249

GCL: Geometric Conservation Law, 17, 213, 415
Germano identity, 254
Global coarsening, 317
Global time stepping, 187
GMRES: Generalised Minimal Residual, 51, 208, 427
Green-Gauss gradient computation, 160, 293
Grid cells, 29
Grid converged solution, 41, 333
Grid generation, 29
Grid lines, 29
Grid nodes, 29
Grid points, 29
Grid refinement, 316
Grid smoothing, 383
Grid vertices, 29
Gridless Method, 40

H-CUSP scheme, 104
Hanging nodes, 32
High Reynolds number model, 245
Horse-shoe vortex, 279
Hybrid grids, 32, 129, 379
Hyperbolic grid generation, 356, 365

IGES: Initial Graphics Exchange Specification, 353
ILU: Incomplete Lower Upper, 51, 210
Implicit operator, 49, 190, 191, 305
Implicit scheme, 190
Implicit time-stepping scheme, 46, 49
Implicit-explicit residual smoothing, 47
Incremental point insertion, 369
Inflow, 8
Integral formulation, 7
Internal energy, 10
Interpolation operator, 306
IRS: Implicit Residual Smoothing, 47, 301
Iterative methods, 50, 191

Jacobi preconditioning, 47
Jacobian coordinate transformation, 37
Jacobian matrix, 419, 421
JST scheme, 96

K-ω model, 55
K-ε model, 55, 241
Kinematic eddy viscosity, 229
Kinematic viscosity coefficient, 15
Kronecker symbol, 431
Krylov subspace, 50, 208

Laplacian operator, 151, 383
LDFSS: Low-Diffusion Flux-Splitting Scheme, 42, 98
Least-squares gradient computation, 162
Left state, 43, 84, 90, 93, 140
Left/right preconditioning, 51, 210
LES: Large-Eddy Simulation, 53, 248
Limiter, 44, 94, 110, 165
Limiter for CUSP scheme, 114
Limiter for TVD scheme, 115
Limiter function, 44, 94, 110, 165
Line-implicit methods, 50
Linear reconstruction, 157
Linear TFI, 362
Linelets, 50
Local time-stepping, 46, 187
Low Reynolds number model, 241
LU-SGS: Lower-Upper Symmetric Gauss-Seidel, 50, 202
LU-SSOR: Lower-Upper Symmetric Successive Overrelaxation, 50, 202
Lumped mass matrix, 45, 182, 190

MAPS: Mach Number-Based Advection Pressure Splitting, 42, 98
Mass matrix, 45, 181, 190, 212, 214
Matrix dissipation scheme, 42, 98, 153
Matrix-free approach, 51, 209
Max-min triangulation, 367
Median-dual control volume, 132, 142

Mesh, 29
Method of lines, 30, 45, 75, 129, 181
Method of weighted residuals, 39
Metric terms, 404
Metrics, 79, 134
Mixed grids, 32, 129, 367, 379
Monotonicity preserving scheme, 43, 110, 165
Morkovin's hypothesis, 53, 232
Multiblock approach, 32, 290
Multigrid cycle, 47
Multigrid level, 47
Multigrid method, 47, 305
Multiphase flow, 22
Multistage scheme, 182
Multistage time-stepping scheme, 46
MUSCL: Monotone Upstream-Centred Schemes for Conservation Laws, 94, 111, 155

Navier-Stokes equations, 17
Newton-Krylov method, 52, 191, 208, 210
Newtonian fluid, 13
Non-linear eddy viscosity, 55, 235
Nonnested grids, 316
Normal-momentum relation, 270
Noslip boundary condition, 57, 275
NURBS: Non-Uniform Rational B-Splines, 378

Order of accuracy, 36, 333
Outflow, 8
Overlapping control volumes, 37, 78, 85, 132
Overrelaxation parameter, 204, 206, 208

Perfect gas, 18
Periodic boundary, 359
Phase angle, 335
Physical space, 29
Piecewise linear prolongation, 319
PNS: Parabolised Navier-Stokes, 23, 409, 418
Point implicit, 186
Prandtl number, 19, 55, 234, 403
Preconditioning, 30, 320
Preconditioning matrix, 321
Pressure sensor, 97
Pressure-based schemes, 30, 320
Prolongation operator, 307
Pseudo-Laplacian, 151

Quadratic reconstruction, 158

RANS: Reynolds-Averaged Navier-Stokes, 53, 231
RCM: Reverse Cuthill-McKee, 195, 210
Real gas, 19, 44
Reconstruction, 133, 139, 145, 150, 154, 160
Reflected cells, 285
Residual, 45, 76, 132, 181
Residual averaging, 47
Residual distribution, 86
Residual smoothing, 47, 301
Restriction operator, 306
Reynolds averaging, 53, 229
Reynolds-stress tensor, 54, 231, 232
Right state, 43, 84, 90, 93, 140
Roe average, 106
Roe matrix, 106
Roe scheme, 43, 106
Rotating frame of reference, 17, 411
Rotation-rate tensor, 228
Rotational periodicity, 287, 289
Rothalpy, 412
Rotor-stator interaction, 380
RST: Reynolds-Stress Transport, 54, 236
Runge-Kutta time-stepping scheme, 46, 182

Scalar dissipation scheme, 96
Search directions, 51, 209
Second viscosity coefficient, 13
Second-order closures, 54, 225
Seed points, 317
Semicoarsening, 306
SGS: Subgrid-Scale stress, 250

Shape functions, 39
Sharp Fourier cut-off filter, 249
Single-stage time-stepping scheme, 46
Singletons, 317
SLIP: Symmetric Limited Positive, 96
Slope limiter, 111
Smagorinsky SGS model, 253
Smooth grid, 29, 131
Solution reconstruction, 139, 145, 150, 154, 160
Solution update, 181
Source term, 76, 131, 185, 190
Spalart-Allmaras model, 55, 238
Spatial averaging, 229
Spatial discretisation, 32, 76, 131
Spectral Element Method, 40
Spectral radius, 97, 153, 187–189
Stability, 331
Stability analysis, 334
Staggered grid scheme, 78
Steger-Warming flux-vector splitting, 195
Steiner point, 372
Stencil, 93
Stiffness, 185, 190
Stokes's hypothesis, 403
Strain-rate tensor, 228
Structured grids, 32, 356
Structured scheme, 75
Subgrid-scale model, 53, 248, 252
Surface forces, 9
Surface grid, 356
Surface grid generation, 379
Sutherland formula, 18
Switched Evolution Relaxation, 211
Symmetric TVD scheme, 43, 108
System matrix, 190, 305

Tensor notation, 228, 431
TFI: Transfinite Interpolation, 356, 359
TFQMR: Transpose-Free Quasi-Minimum Residual, 51, 208
Thermal conductivity coefficient, 11
Thermal diffusivity coefficient, 7, 11
Time averaging, 229
Time step, 186
Time-stepping operator, 335
Tophat filter, 249
Total energy, 10, 411
Total enthalpy, 12, 412
Translational periodicity, 287
Truncation error, 36, 332
TSL: Thin Shear Layer, 23, 119, 199, 416
Turbulence modelling, 53, 225
Turbulent dissipation rate, 236
Turbulent eddy viscosity, 55, 233
Turbulent heat-flux vector, 54, 233, 234
Turbulent kinetic energy, 231, 232
Turbulent Prandtl number, 55, 234
Turbulent thermal conductivity coefficient, 55, 234
TVD: Total Variation Diminishing, 42, 43, 108
Two-equation models, 241

UIRS: Upwind Implicit Residual Smoothing, 47, 303
Unit normal vector, 6, 16, 77, 79–81, 134, 135, 137, 140, 144, 244, 412, 414
Unsteady flows, 49, 212
Unstructured grids, 32, 34, 367
Unstructured scheme, 129
Upwind prolongation, 310, 314, 316
Upwind restriction, 310, 316
Upwind scheme, 42, 154
Upwind TVD scheme, 43, 108, 109

V-Cycle, 308
Validation, 332
Van Albada limiter, 111
Van Leer's flux-vector splitting, 99
Variational principle, 39
Vector of convective fluxes, 16
Vector of viscous fluxes, 16
Verification, 332
Virtual edges, 164

Viscous flux Jacobian, 199, 421
Viscous fluxes, 16, 116, 169
Viscous stress tensor, 9, 228
Viscous stresses, 13
Visibility criterion, 378
Volume agglomeration, 317
Volume grid, 356
Von Neumann stability analysis, 334
Voronoj diagram, 368
Vortex correction, 280, 281

W-Cycle, 308
Wall functions, 245
Weak formulation, 39
Weak solutions, 38

COMPUTATIONAL FLUID DYNAMICS: PRINCIPLES AND APPLICATIONS

Computational Fluid Dynamics (CFD) is an important design tool in engineering and also a substantial research tool in various physical sciences.

The objective of this book is to provide a solid foundation for understanding the numerical methods employed in today's CFD and to raise awareness of modern CFD codes through hands-on experience. The book will be an essential reference work for engineers and scientists starting to work in the field of CFD or those who apply CFD codes.

The accompanying CD-ROM contains the sources of 1-D and 2-D Euler solvers as well as grid generators.

Chapters

1 Introduction
2 Governing Equations
3 Principles of Solution of the Governing Equations
4 Spatial Discretisation: Structured Finite Volume Schemes
5 Spatial Discretisation: Unstructured Finite Volume Schemes
6 Temporal Discretisation
7 Turbulence Modelling
8 Boundary Conditions
9 Acceleration Techniques
10 Consistency, Accuracy and Stability
11 Principles of Grid Generation
12 Description of the Source Codes

ISBN 0 08 043009 0

Front cover image courtesy of
O. Brodersen, DLR, Germany

9 780080 430096